Н. В. Морзе, О. В. Барна, В. П. Вембер

ІНФОРМАТИКА

Підручник для 8 класу
загальноосвітніх навчальних закладів

Рекомендовано Міністерством освіти і науки України

Київ
Оріон
2016

УДК 004(075.3)
ББК 32.973я721
М79

Морзе Н. В.

М79 Інформатика : підруч. для 8 кл. загальноосвіт. навч. закладів / Н. В. Морзе, О. В. Барна, В. П. Вембер. — К. : УОВЦ «Оріон», 2016. — 240 с.

ISBN 978-617-000-000-0.

УДК 004(075.3)
ББК 32.973я721

ISBN 978-617-000-000-0

© Н. В. Морзе, О. В. Барна, В. П. Вембер, 2016
© УОВЦ «Оріон», 2016

ДОРОГІ ВОСЬМИКЛАСНИКИ!

Цей підручник допоможе вам продовжити вивчення інформатики. Він складається із семи розділів, що містять теми, які ви розглядатимете на одному, двох або більше уроках. Ви зрозумієте, як кодуються дані для їх опрацювання за допомогою комп'ютера, навчитеся знаходити відмінності між різними складовими комп'ютера, пристроями та програмним забезпеченням. Продовжите удосконалювати свої знання й уміння з опрацювання текстових даних у середовищі текстового процесора та числових даних за допомогою табличного процесора. Опануєте нові програми для роботи з об'єктами мультимедіа. На сторінках підручника ви знайдете опис мов програмування *Python* і *Free Pascal* та приклади створення відповідних програм, а за допомогою середовищ програмування, що підтримують ці мови, навчитеся розробляти програмні проекти, які можуть стати у пригоді в навчанні, побуті й розвагах та, можливо, сприяти майбутньому вибору професії. У підручнику багато завдань і вправ, у тому числі — компетентнісних задач і проектів, розв'язуючи та виконуючи які, ви набуватимете нових знань й умінь.

Готуючись до уроку, зверніть увагу на перелік питань, які ви вже вивчали в курсі інформатики в попередніх класах, позначений заголовком

ПРИГАДАЙТЕ:

Намагайтеся відновити в пам'яті вивчене, скористайтеся записами в робочих зошитах чи відомостями з Інтернету та будьте готовими використати свої знання для вивчення нових тем, набуття нових умінь та розвитку компетентностей.

Будуйте власну стратегію навчання, здійснюйте рефлексію та оцінювання, використовуючи карту знань теми

ВИ ДІЗНАЄТЕСЯ:

Адже в карті знань зазначені найважливіші питання, які буде розглянуто в темі.

У підручнику ви будете використовувати такі рубрики.

ВИВЧАЄМО

Матеріали рубрики **Вивчаємо** доцільно читати напередодні уроку, щоб разом з однокласниками й учителем обговорити незрозуміле та поділитись вивченим.

Найважливіші означення з цієї теми виділено та позначено так.

ДІЄМО

Виконуючи вправи з рубрики **Діємо**, ви навчитеся кодувати дані, створювати мультимедійні проекти, опрацьовувати текстові та числові дані, розробляти в середовищах програмування проекти для опрацювання даних різного типу.

ДОСЛІДЖУЄМО

Рубрика **Досліджуємо** містить завдання, у яких потрібно експериментувати, самостійно шукати відповіді, відкривати для себе нові можливості.

«Фаворитні» завдання, що містять позначки , , якраз для вас — тих, що в майбутньому планують бути успішними.

Рубрика **Обговорюємо** спрямовує вас до списку питань, які можна обговорити колективно або в малих групах. Їх можна знайти у папці *Обговорюємо* або за посиланням, поданим відповідним QR-кодом.

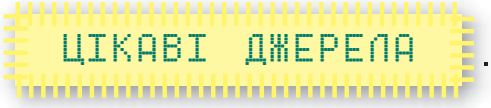
Сподіваємось, ви навчитеся планувати свою навчальну роботу, мислити логічно, шукати незвичні розв'язки завдань, робити висновки та самостійно приймати рішення. Для цього виконуйте вправи із рубрики **Працюємо самостійно**.

Також важливо вміти доводити свої думки під час співпраці в парах та групах або визнавати хибність своїх міркувань на користь більш логічних і доказових. Для цього призначена рубрика **Працюємо в парах.**

Співпраця, уміння вчитись одне в одного, підтримка у практичній діяльності — запорука успіху в майбутньому.

Не обминайте закладок

. Адже вивчення інформатики сприяє всебічному розвитку, формуванню вміння самостійно навчатись.

Обов'язково розгляньте посилання, записані в закладці 

.

Швидко ввести потрібну адресу можна за допомогою QR-коду. Тут ви отримаєте багато корисних відомостей та інструментів для навчання, які доступні в мережі Інтернет.

Бажаємо вам успіху!

РОЗДІЛ 1

КОДУВАННЯ ДАНИХ

1. КОДУВАННЯ ДАНИХ

ПРИГАДАЙТЕ:

- як пов'язані між собою поняття інформація, повідомлення, дані;
- способи подання повідомлень;
- основні інформаційні процеси: пошук, передавання, опрацювання, зберігання повідомлень;
- особливості опрацювання повідомлень.

ВИ ДІЗНАЄТЕСЯ:

- що таке кодування повідомлень;
- які особливості має двійкове кодування;
- як кодуються в комп'ютері текстові повідомлення;
- для чого використовуються таблиці кодування символів;
- як визначити довжину двійкового коду повідомлення.

ВИВЧАЄМО

1. Що таке кодування повідомлень?

Ви знаєте, що інформація передається за допомогою повідомлень, які можна подати текстовим, графічним, звуковим способами, а також відео, жестами та декількома способами одночасно — комбінованим повідомленням.

Для подання усних повідомлень ми використовуємо мову. А будь-яке письмове повідомлення складається з набору різних знаків (символів), які належать певному алфавіту. Наприклад, для подання числових даних використовують алфавіт, що складається з 11 знаків: цифр 0–9 і десяткової коми для відокремлення цілої та дробової частини в десятковому дробі.

Людина постійно отримує, шукає, зберігає, опрацьовує й передає повідомлення. При цьому вона часто застосовує різні пристрої. Використання пристроїв зумовлює необхідність подання повідомлень за допомогою спеціальних знаків.

Код — це система правил для перетворення повідомлень, що містять текст, звук, зображення, жести тощо.

Код Морзе	Штрих-код	QR-код
. . . — — — . . .		

Кодування повідомлень — це процес перетворення за певними правилами одного подання набору даних в інший. Під час кодування повідомлення відбувається зміна вигляду повідомлення без зміни його змісту.

Мал. 1.1

У процесі обміну повідомленнями ми здійснюємо дві операції: кодування та декодування. Перша пов'язана з переходом від вихідної форми подання повідомлення у форму, яка є зручною для збереження, передавання або опрацювання. А друга — зі зворотним переходом до вихідного подання повідомлення.

Процес отримання та передавання повідомлень можна зобразити у вигляді схеми (мал. 1.1).

Кодування й декодування належать до процесу опрацювання повідомлень, який можна здійснювати за допомогою комп'ютера.

Останнім часом дедалі частіше для кодування даних почали використовувати QR-коди.

QR-код — квадратна картинка, в якій закодовано деякі відомості:

- звичайний текст;
- адреса веб-сторінки в Інтернеті;
- номер телефону;
- координати місця на карті;
- персональна візитна картка особи тощо.

Виглядає QR-код як квадратне зображення, на якому зазвичай розташовані три великі квадрати у кутах. Вони слугують своєрідними орієнтирами для програм-зчитувачів. За їх допомогою визначається рівень нахилу та деформація пропорцій. Решта фігур — це власне і є зашифровані відомості. На відміну від класичного штрих-коду, QR-код може містити в собі доволі великі обсяги даних.

Спеціальний вигляд QR-кодів забезпечує зчитування та розкодування даних за допомогою сучасних пристроїв, обладнаних камерами, наприклад, мобільних телефонів. Достатньо навести камеру телефону на код, й одразу можна отримати доступ до його вмісту.

З розповсюдженням потужних мобільних телефонів, обладнаних вбудованими камерами, QR-коди почали широко використовувати у світі.

Для зчитування QR-кодів за допомогою мобільного телефону на ньому потрібно встановити спеціальну програму (мал. 1.2).

QR-коди створюють за допомогою спеціальних програм, які можна завантажити на комп'ютер, або використовують спеціальні сайти в Інтернеті. Наприклад, за допомогою онлайнового сервісу для генерування QR-коду створено QR-код його адреси *http://ua.qr-code-generator.com/*

Цікаво

QR-коди (від англ. *Quick Response* — швидка реакція, швидка відповідь) були розроблені в 1994 р. японською компанією Denso-Wave. В Японії QR-коди набули широкого розповсюдження ще на початку 2000-х років: їх розміщують на упаковці товарів, друкують у буклетах, використовують у рекламі, довідниках, іграх тощо.

Мал. 1.2

2. Які особливості має двійкове кодування?

Повідомлення можуть кодуватися різними способами. Вибір способу кодування залежить від виду повідомлення, яке необхідно кодувати: текст, число, графічне зображення або звук чи відео. Для опрацювання за допомогою комп'ютера повідомлення подаються (кодуються) у вигляді послідовностей електричних або магнітних сигналів двох видів. Кожний сигнал одного виду умовно позначають цифрою 0, а іншого виду — 1.

Кодування повідомлень із використанням двох сигналів називається **двійковим**. Набір даних, отриманий у результаті двійкового кодування, називається **двійковим кодом.**

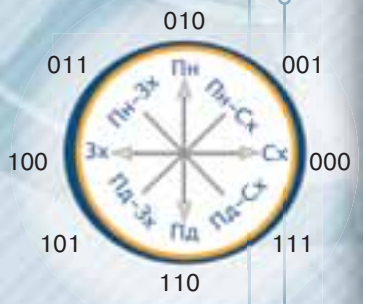
Цифра 0 або 1 у двійковому коді повідомлення має назву **біт** (англ. *binary digit* — двійкова цифра).

Однією з двох цифр 0 або 1 можна закодувати, наприклад:

- правильність твердження: неправильно (0) або правильно (1);
- стан вимикача: вимкнено (0) або ввімкнено (1) тощо.

Із двох бітів можна скласти 4 ($4 = 2^2$) коди (00, 01, 10 і 11). Ними можна закодувати, наприклад, чотири чверті координатної площини: 00 — ліва верхня; 01 — права верхня; 10 — ліва нижня; 11 — права нижня.

Із трьох бітів можна скласти вже 8 ($8 = 2^3$) кодів (000, 001, 010, 011, 100, 101, 110, 111). Ними можна закодувати, наприклад, сторони горизонту (мал. 1.3). Із чотирьох бітів можна скласти $2^4 = 16$ кодів, з п'яти — $2^5 = 32$ коди і т. д. З восьми бітів можна скласти $2^8 = 256$ кодів, і цієї кількості кодів достатньо, щоб закодувати всі літери англійського та українського (або якогось іншого) алфавітів, арабські цифри, розділові знаки, знаки арифметичних дій, а також деякі інші символи.



Мал. 1.3

Послідовність із восьми бітів має назву **байт.**

1 байт = 8 біт

Таблиця степенів числа 2 демонструє кількість різних повідомлень, які можна закодувати за допомогою відповідної кількості бітів:

i	0	1	2	3	4	5	6	7	8	9	10
$N = 2^i$	1	2	4	8	16	32	64	128	256	512	1024

Цікаво

За допомогою формули Хартлі $2^i = N$ можна визначити можливу кількість різних повідомлень N, якщо i — кількість бітів у коді повідомлення.

3. Як кодуються в комп'ютері текстові повідомлення?

Традиційно для кодування одного символу використовується двійковий код, довжина якого дорівнює 1 байту, тобто 1 символ = 1 байт = = 8 біт. Тоді можна обчислити, яку кількість різних символів можна закодувати: $N = 2^8 = 256$. Така кількість символів є цілком достатньою для

подання текстових повідомлень, включаючи великі та малі літери українського й латинського алфавітів, цифри, знаки, графічні символи тощо. Кодування полягає в тому, що кожному символу ставиться у відповідність унікальний десятковий код від 0 до 255 або відповідний йому двійковий код від 00000000 до 11111111.

Користувач натискає на клавіатурі клавішу із символом, і в комп'ютер надходить певна послідовність із восьми електричних імпульсів (двійковий код символу). Код символу зберігається в оперативній пам'яті комп'ютера, де займає один байт. У процесі виведення символу на екран комп'ютера проводиться зворотний процес — декодування, тобто перетворення коду символу в його зображення.

Довжина двійкового коду текстового повідомлення — це кількість бітів чи байтів у двійковому коді цього повідомлення.

При такому кодуванні один рядок тексту цього підручника має середню довжину двійкового коду приблизно 50 байт, одна сторінка — приблизно 2000 байт, а весь підручник (240 с.) — приблизно 480 000 байт.

Для позначення довжин двійкового коду повідомлень використовують і більші одиниці вимірювання, назви яких, згідно з Міжнародною системою одиниць (СІ), утворюються за допомогою префіксів кіло-, мега-, гіга-, тера- і т. д. Історично склалося так, що ці префікси (кіло-, мега-, гіга-, тера-) в інформатиці трактувалися по-іншому, не так, як, наприклад, у математиці, і мають у своїй основі степені числа 2, а саме:

1 Кб (кілобайт) = 2^{10} байт = 1024 байт;

1 Мб (мегабайт) = 2^{10} Кб = 2^{20} байт = 1 048 576 байт;

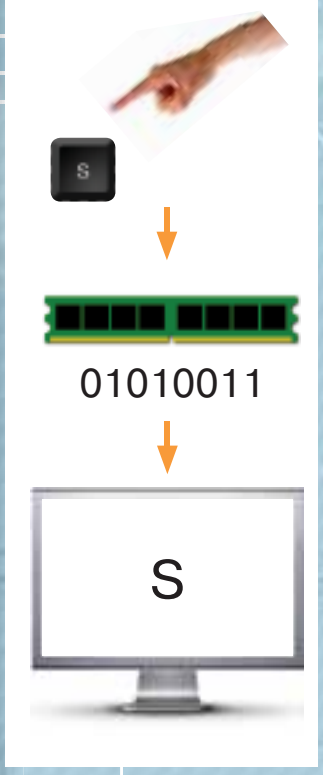
1 Гб (гігабайт) = 2^{10} Мб = 2^{20} Кб = 2^{30} байт;

1 Тб (терабайт) = 2^{10} Гб = 2^{20} Мб = 2^{30} Кб = 2^{40} байт.

Щоб подати біти в байтах, треба число бітів поділити на 8. Наприклад: 32 біти — це 4 байти. Щоб подати байти в кілобайтах, треба число байтів поділити на 1024. Наприклад: у 2048 байтах буде 2 Кб. І так далі за наступними одиницями вимірювання.

Щоб подати байти в бітах, треба число байтів помножити на 8. Наприклад: у 3 байтах буде 24 біти.

Щоб подати кілобайти в байтах, треба число кілобайтів помножити на 1024. Наприклад: у 3 кілобайтах буде 3072 байти і відповідно 24 576 біт.



ДІЄМО

Вправа 1. Довжина двійкового коду тексту.

Завдання. Книжка містить 150 сторінок, на кожній сторінці — 40 рядків, у кожному рядку 60 символів (включаючи пропуски). Знайдіть довжину двійкового коду тексту книжки, якщо для кодування кожного символу використано 8 біт.

Розв'язання. Обчислимо кількість символів у книжці:

60 · 40 · 150 = 360 000 символів.

Оскільки довжина двійкового коду 1 символу — 8 біт =1 байт, довжина двійкового коду книжки дорівнює 360 000 байт.

Подамо байти в більших одиницях: 360 000 байт : 1024 = 351,56 Кб.

351,56 Кб : 1024 = 0,34 Мб.

Таблиця 1.1

Код	Символ	Код	Символ	Код	Символ	Код	Символ
32	про-біл	56	8	80	P	104	h
33	!	57	9	81	Q	105	i
34	«	58	:	82	R	106	j
35	#	59	;	83	S	107	k
36	$	60	<	84	T	108	i
37	%	61	=	85	U	109	m
38	&	62	>	86	V	110	n
39	‘	63	?	87	W	111	0
40	(	64	@	88	X	112	p
41	)	65	A	89	У	113	q
42	*	66	B	90	Z	114	r
43	+	67	C	91	[	115	s
44	,	68	D	92	\	116	t
45	-	69	E	93	]	117	u
46	.	70	F	94	^	118	v
47	/	71	G	95	_	119	w
48	0	72	H	96	`	120	x
49	1	73	I	97	a	121	y
50	2	74	J	98	b	122	z
51	3	75	K	99	c	123	{
52	4	76	L	100	d	124	|
53	5	77	M	101	e	125	}
54	6	78	N	102	f	126	~
55	7	79	O	103	g	127	DEL

4. Для чого використовуються таблиці кодування символів?

Натиснення клавіші на клавіатурі приводить до того, що сигнал надсилається в комп’ютер у вигляді двійкового числа, яке зберігається в кодовій таблиці. Кодова таблиця встановлює відповідність між символами та їх двійковими кодами для подання текстових даних у комп’ютері. Для того щоб весь світ однаково кодував текстові дані, потрібні єдині таблиці кодування.

У всьому світі за стандарт прийнято таблицю ASCII (*American Standard Code for Information Interchange* — Американський стандартний код для обміну повідомленнями). Створена в 1963 р. система кодування ASCII передбачала кодування 128 символів, коди яких складалися із 7 біт ($2^7 = 128$). З часом кодування було розширене до 256 символів ($2^8 = 256$), при цьому коди перших 128 символів не змінилися.

Таблиця кодування ASCII (табл. 1.1) складається з базової (значення кодів від 0 до 31) і розширеної (значення кодів від 32 до 255) таблиць. Коди з 33 по 127 є інтернаціональними й відповідають символам латинського алфавіту, цифрам, знакам арифметичних операцій та знакам пунктуації. Коди зі 128 по 255 є національними, тобто в національних кодуваннях одному й тому самому коду відповідають різні символи.

У світі існують й інші системи кодування. У різних таблицях кодування одні й ті самі символи можуть мати різні коди. Останнім часом серед таблиць кодування, які містять українські літери, найпоширенішими є KOI8-U і Windows-1251. Довжина коду кожного символу в них — 1 байт.

Широкого поширення набув міжнародний стандарт Unicode — *Unicode Consortium* (UTF 32, UTF 16 і UTF 8), який відводить на кожний символ не один байт, а два, тобто 16 біт, тому за його допомогою можна закодувати не 256, а $2^{16} = 65\,536$ різних символів.

ДІЄМО

Вправа 2. Тексти в різних системах кодування.

Завдання. Обчисліть довжину двійкового коду в різних системах кодування для текстів, поданих у клітинках електронної таблиці *Тексти*.

Розв’язання.

1. У власній структурі папок створіть папку *Кодування*.
2. Відкрийте файл *Тексти*, збережений у папці *Кодування даних*.
3. Введіть у клітинку *B4* формулу обчислення кількості символів у тексті: =LEN(A4) (=ДЛСТР(A4) для середови-

	A	B	C	D	E
1			Код		
2		Кількість символів	Windows-1251	ASCII	Unicode
3	Прислів'я		8	8	16
4	Не кажи — не вмію, а кажи — навчусь!	36	288	288	576
5	Не святі горшки ліплять.				
6	Мудрим ніхто не вродився, а навчився.				

Мал. 1.4

ща з інтерфейсом російською мовою). Переконайтеся, що вбудована в табличний процесор функція правильно визначає кількість символів.

4. Виконайте обчислення в клітинках четвертого рядка так, щоб отримати результат, як на малюнку 1.4.
5. Виконайте обчислення в клітинках п'ятого та шостого рядків.
6. Збережіть файл із тим самим іменем у папці *Кодування* у власній структурі папок.

ЦІКАВІ ДЖЕРЕЛА

Стаття про системи кодування у Вікіпедії:

https://uk.wikipedia.org/wiki/ASCII

5. Як визначити довжину двійкового коду повідомлення?

Щоб обчислити довжину двійкового коду повідомлення, треба кількість символів у тексті помножити на кількість бітів, потрібних для кодування одного символу. Наприклад: двійкове число 01010111 займає в пам'яті 8 біт. Якщо його записати у вигляді тексту в кодуванні ASCII, довжина коду буде 8 байт, або 64 біти, оскільки кожний символ кодується за допомогою 8 біт. Довжина двійкового коду цього ж тексту в кодуванні Unicode становитиме 16 байт, або 128 біт.

Не слід забувати, що пропуски треба вважати за символи, оскільки вони також набираються на клавіатурі, мають код і зберігаються в пам'яті.

ДІЄМО

Вправа 3. Перекодування.

Завдання. Автоматичний пристрій здійснив перекодування текстового повідомлення довжиною 48 символів, спочатку записаного в 7-бітному коді ASCII, у 16-бітне кодування Unicode. Визначте, на скільки при цьому збільшилася довжина двійкового коду повідомлення.

Розв'язання. Зміна кодування із 7 біт на 16 біт збільшує довжину коду кожного символу на 16 – 7 = 9 біт. Оскільки повідомлення містить 48 символів, то його обсяг збільшився на 48 · 9 = 432 біти. Подамо біти в байтах: 432 : 8 = 54 байти.

Вправа 4. Модем.

Завдання. Визначте, скільки часу модем, що працює зі швидкістю 1200 біт/с, буде передавати дані десяти сторінок тексту із 40 рядків по 80 символів у рядку.

Розв'язання. Обчислимо загальну кількість символів на сторінці. Це 40 · 80 = 3200 символів.

Оскільки в кодуванні ASCII один символ має довжину коду 1 байт, загальна довжина двійкового коду даних на сторінці — 3200 байт. Виразимо 3200 байт у бітах, оскільки швидкість модема вказана в бітах за секунду. Отримаємо 3200 байт = 25 600 біт.

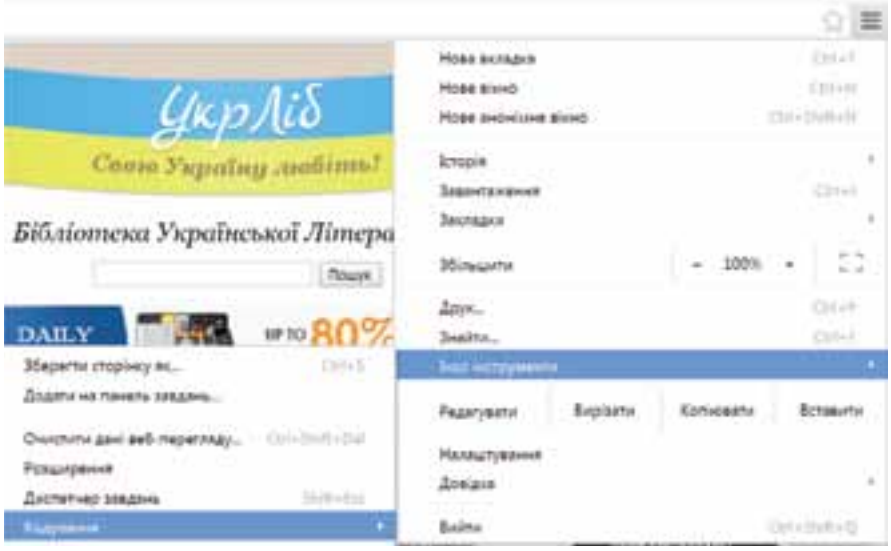
Розділимо 25 600 біт на 1200 біт/с, отримаємо 21,33 с. Таким чином, 10 сторінок тексту буде передано модемом за 213,3 с, або 3 хв 33,3 с.

ДОСЛІДЖУЄМО

Вправа 5. Кодування веб-сторінок.

Завдання. Дослідіть, як зміняться текстові дані на головній сторінці веб-порталу *Бібліотека української літератури* (*http://www.ukrlib.com.ua/*), якщо змінити систему кодування, що використана при її створенні.

1. Відкрийте вікно браузера, наприклад *GoogleChrome*. У полі адреси введіть адресу веб-порталу *Бібліотека української літератури* (*http://www.ukrlib.com.ua/*). Перегляньте, чи правильно відображаються текстові написи на сторінці.
2. Натисніть кнопку налаштування та керування браузером. Оберіть у меню, що відкрилося, вказівку *Інші інструменти*/*Кодування* (мал. 1.5).



Мал. 1.5

3. Послідовно обирайте систему кодування з меню, що відкрилося. Зробіть висновок, яка із систем кодування, окрім системи Windows-1251, правильно відображає вміст веб-сторінки.

ОБГОВОРЮЄМО

Обговоріть запитання, що містяться у файлі *Тема 1*, який зберігається у папці *Обговорюємо*. Також відкрити ці запитання можна за допомогою QR-коду.

ПРАЦЮЄМО В ПАРАХ

1. Створіть у табличному процесорі таблицю переведення одиниць довжини двійкового коду повідомлення в такому вигляді:

Кількість символів двійкового коду, біт	байт	Кбайт	Мбайт	Гбайт	Тбайт

Запропонуйте одне одному та перевірте на практиці інші таблиці переведення, наприклад, кілобайтів у біти, байти, мегабайти, гігабайти, терабайти. Обговоріть, як при цьому будуть змінюватись формули у клітинках електронної таблиці.

2. Запропонуйте одне одному текстове повідомлення, закодоване за допомогою однієї із систем кодування. Один подає придумане повідомлення, використовуючи таблицю Unicode, інший — Windows-1251. Перевірте правильність кодування, для цього обміняйтесь кодами та декодуйте повідомлення. Визначте довжину двійкового коду кожного з повідомлень. Зробіть висновок.

ПРАЦЮЄМО САМОСТІЙНО

1. Визначте довжину двійкового коду слова із 24 символів у кодуванні Unicode.
2. Довжина двійкового коду текстового повідомлення становить 8192 біти. Виразіть це значення в кілобайтах.
3. Довжина двійкового коду текстового повідомлення становить 2 097 152 байти. Виразіть це значення в мегабайтах.
4. Довжина двійкового коду текстового повідомлення, підготовленого за допомогою комп'ютера, становить 3,5 Кб. Скільки символів містить цей текст? Чи достатньо даних має задача, щоб можна було дати однозначну відповідь?
5. Автоматичний пристрій здійснив перекодування текстового повідомлення українською мовою, спочатку записаного в 16-бітному коді Unicode, у 8-бітне кодування KOI-8. При цьому довжина двійкового коду повідомлення зменшилася на 480 біт. Скільки символів містить повідомлення?
6. Відомо, що на кожній сторінці документа 128 рядків, що містять по 48 символів кожний. Скільки сторінок у документі, якщо довжина його двійкового коду становить 720 Кб, за умови, що кожний символ кодувався 2-байтовим кодуванням Unicode?

ЦІКАВІ ДЖЕРЕЛА

Відеоурок «Кодування текстової інформації»:
http://www.youtube.com/watch?v=3BDE0oxevUQ

2. ПРАКТИЧНА РОБОТА 1

РОЗВ'ЯЗУВАННЯ ЗАДАЧ НА ВИЗНАЧЕННЯ ДОВЖИНИ ДВІЙКОВОГО КОДУ ТЕКСТОВИХ ДАНИХ

ПРИГАДАЙТЕ

- як визначати довжину двійкового коду текстового повідомлення.

СТВОРІТЬ

У власній структурі папок створіть папку *Практична робота 1.*

ПАМ'ЯТАЙТЕ

Під час виконання практичних завдань пам'ятайте про правила безпеки життєдіяльності при роботі з комп'ютером!

1. Довжина двійкового коду тексту, набраного на комп'ютері з використанням кодування Unicode (кожний символ кодується 16 бітами), — 4 Кб. Визначте кількість символів у тексті. Для розв'язування задачі складіть електронну таблицю, обчислення виконайте за допомогою формул. Результат збережіть у файлі з іменем *Завдання 1* у папці *Практична робота 1* власної структури папок (*1 бал*).

2. Автоматичний пристрій здійснив перекодування інформаційного повідомлення українською мовою, спочатку записаного в 16-бітному коді Unicode, у 8-бітне кодування Windows-1251, при цьому довжина двійкового коду повідомлення становила 60 байт. Визначте довжину двійкового коду повідомлення до перекодування. Для розв'язування задачі складіть електронну таблицю, обчислення виконайте за допомогою формул. Результат збережіть у файлі з іменем *Завдання 2* в папці *Практична робота 1* власної структури папок (*2 бали*).

3. Повідомлення займає 3 сторінки й має довжину двійкового коду 7950 байт. Визначте, скільки рядків на сторінці, якщо в кожному рядку 25 символів і використане кодування Unicode. Для розв'язування задачі складіть електронну таблицю, обчислення виконайте за допомогою формул. Результат збережіть у файлі з іменем *Завдання 3* в папці *Практична робота 1* власної структури папок (*2 бали*).

4. Подайте текст «Любіть Україну, як сонце, любіть!» у вигляді коду, використовуючи таблицю кодів міжнародного стандарту Unicode. Таблицю кодів знайдіть в Інтернеті. Результат подайте у вигляді текстового документа і збережіть у файлі *Завдання 4* в папці *Практична робота 1* власної структури папок (*3 бали*).

5. Повідомлення, набране на комп'ютері з використанням системи кодування Windows-1251, містить 1536 символів. Визначте довжину двійкового коду повідомлення в кілобайтах (*2 бали*).

6. Довжина двійкового коду повідомлення становить 0,3 Мб. Виразіть це значення в кілобайтах (*1 бал*).

7. Знайдіть орієнтовну довжину двійкового коду завдань практичної роботи. Подайте отримане значення в різних одиницях вимірювання (*3 бали*).

РОЗДІЛ 2

АПАРАТНО-ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ КОМП'ЮТЕРА

3. АПАРАТНЕ ЗАБЕЗПЕЧЕННЯ КОМП'ЮТЕРА

ПРИГАДАЙТЕ:

- складові сучасного комп'ютера;
- які пристрої використовують для введення, виведення, зберігання даних;
- види сучасних персональних комп'ютерів.

ВИ ДІЗНАЄТЕСЯ:

- як технічний прогрес вплинув на розвиток обчислювальної техніки;
- яку архітектуру мають сучасні комп'ютери;
- чи може комп'ютер працювати без процесора;
- які пристрої комп'ютера розміщуються всередині системного блока;
- що і як можна зберігати в пам'яті комп'ютера;
- які особливості мають пристрої введення та виведення даних;
- які пристрої входять до складу мультимедійного обладнання;
- як обрати найкращий комп'ютер для роботи.

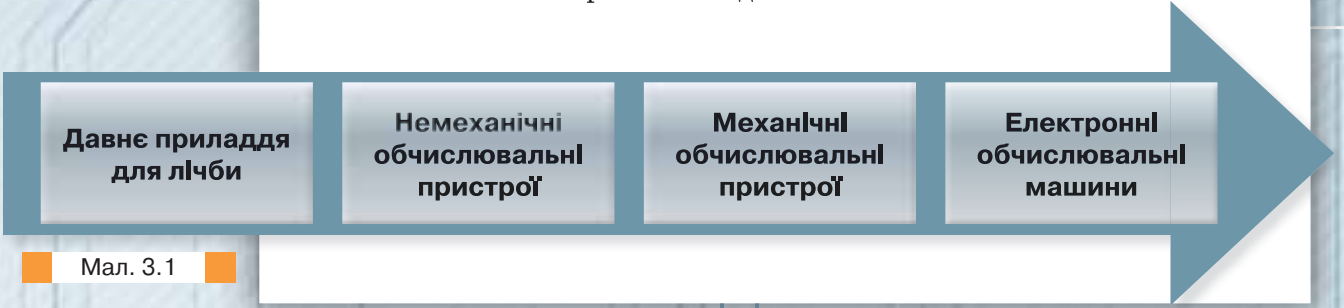
ВИВЧАЄМО

1. Як технічний прогрес вплинув на розвиток обчислювальної техніки?

Комп'ютерна техніка використовується для обчислень й опрацювання інформаційних об'єктів різного типу. Першим приладдям для лічби були, ймовірно, зарубки, вузлики, лічильні палички. Розвиваючись, це приладдя ставало складнішим, і з часом з'явилися нові пристрої: абак (рахівниця), логарифмічна лінійка, механічний арифмометр, електронний комп'ютер. Засоби для обчислень постійно змінювались і пройшли кілька етапів розвитку (мал. 3.1).

Обчислювальна техніка поступово стала використовуватися не лише для обчислень, а й для вирішення інших завдань, наприклад, для автоматизації різних процесів, використання електронних засобів зв'язку, контролю обладнання, виконання офісних завдань, комп'ютерних ігор, навчання тощо. Кожна галузь, у свою чергу, запровадила додаткові вимоги до комп'ютерного обладнання, яке постійно розвивається відповідно до цих вимог.

Удосконалення комп'ютерів постійно відбувається в кількох напрямах. По-перше, змінюють або ж використовують нові основні елементи, з яких виготовляють комп'ютер, — змінюється елементна база комп'ютерів. По-друге, змінюється програмне забезпечення. Крім того, вдосконалюються пристрої введення та виведення даних та організація і взаємозв'язок його окремих складових.



Мал. 3.1

Початком ери комп'ютерів вважають 1945–1946 рр., коли американські вчені Проспер Еккерт і Джон Моучлі сконструювали в Пенсильванському університеті (США) першу ЕОМ «ENIAC» (від англ. — *Electronic Numerical Integrator and Calculator*). ENIAC мала 1800 електронних ламп та 150 000 електромеханічних реле. Зрозуміло, що ця машина була дуже громіздкою, складною в управлінні (щоб змінити програму, необхідно було перепаювати схему), ненадійною в роботі, мала низку інших недоліків. ENIAC відносять до першого покоління електронних обчислювальних машин (ЕОМ).

Під **поколіннями ЕОМ** розуміють усі типи й моделі електронних обчислювальних машин, розроблені різними конструкторськими колективами, але побудовані за одними й тими самими науковими і технічними принципами. Кожне наступне покоління визначається новими електронними елементами, технологія виготовлення яких є принципово іншою, обчислювальними можливостями, швидкодією та іншими властивостями.

Покоління ЕОМ (англ. — *computer generation*) — один із класів у класифікації обчислювальних систем за ступенем розвитку апаратних і програмних засобів.

Зазвичай вирізняють чотири покоління електронної обчислювальної техніки (табл. 3.1).

Швидкого розповсюдження комп'ютери набули, починаючи з третього покоління.

Таблиця 3.1

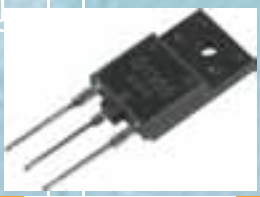
Покоління ЕОМ	Роки	Електронні елементи	Швидкодія
I	1950–1960	Електровакуумні лампи (ENIAC, МЕСМ)	10–20 тис. оп./с
II	1960–1965	Транзистори	100–500 тис. оп./с
III	1965–1970	Інтегральні схеми	1 млн оп./с
IV	з 1970	Мікропроцесори	сотні млн оп./с

У нашій країні розробки ЕОМ почалися також у 1940-х роках. У 1951 р. в Києві під керівництвом професора Сергія Олексійовича Лебедєва (1902–1974) було введено в експлуатацію ЕОМ, яка дістала назву «МЭСМ» (рос. — *Малая электронно-счётная машина*).

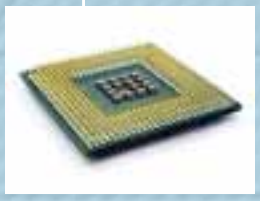
Значний внесок у розвиток вітчизняної комп'ютерної техніки зробив Віктор Михайлович Глушков (1923–1982) — засновник наукової школи кібернетики, автор фундаментальних праць із кібернетики, штучного інтелекту, теорії цифрових автоматів, з питань застосування кібернетичних методів в економіці.

Починаючи з 1950-х років обчислювальна техніка почала бурхливо розвиватися і за кордоном, і в Україні.

Електровакуумна лампа


Транзистор

Інтегральні схеми


Мікропроцесор

С. О. Лебедєв

В. М. Глушков

ДІЄМО

Вправа 1. Історія розвитку обчислювальної техніки.
Завдання. Доповніть таблицю у файлі *Покоління комп'ютерів*, що збережений у папці *Забезпечення комп'ютера*, зображеннями комп'ютерів кожного з поколінь та їх елементної бази, відомостями про видатних людей, причетних до розвитку кожного з поколінь ЕОМ.

1. Створіть папку *Комп'ютери та програми* у власній структурі папок.
2. Відкрийте файл *Музеї комп'ютерної техніки*, збережений у папці *Забезпечення комп'ютера,* та оберіть одне з наведених посилань. Знайдіть зображення комп'ютерів кожного з поколінь, їх елементної бази та відомості про видатних людей, причетних до розвитку кожного з поколінь.
3. Відкрийте файл *Покоління комп'ютерів*, збережений у папці *Забезпечення комп'ютера*. Додайте до клітинок таблиці знайдені відомості.
4. Збережіть файл з іменем *Покоління комп'ютерів_Прізвище* в папці *Комп'ютери та програми* у власній структурі папок.

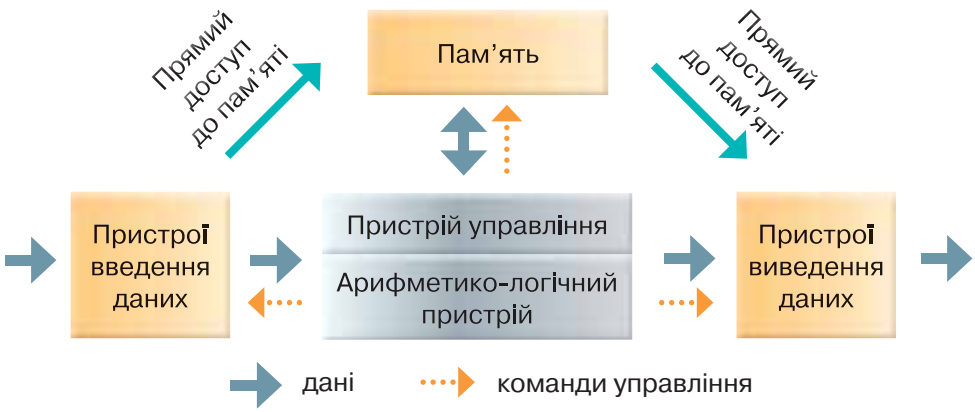
2. Яку архітектуру мають сучасні комп'ютери?

Сучасний стаціонарний комп'ютер може мати такі складові: системний блок, монітор, клавіатуру та мишу, акустичну систему, принтер, сканер тощо. Сьогодні використовують й інші види комп'ютерів: ноутбуки та нетбуки, планшетні та кишенькові. У таких комп'ютерах системний блок, монітор і часто інші пристрої об'єднані в один. Попри те, що сфери застосування сучасних комп'ютерів практично не обмежені, в основу їх роботи покладено єдині принципи роботи. Вони визначають загальну структуру, без урахування особливостей тих чи інших моделей, і відображають основні зв'язки між пристроями комп'ютера, потоки даних, які циркулюють між ними, та принципи їх опрацювання. Уніфікація архітектури ПК забезпечує їх сумісність із точки зору користувача.

Архітектура ПК — принципи роботи та взаємодії основних пристроїв комп'ютера: процесора, внутрішньої й зовнішньої пам'яті та пристроїв введення й виведення даних.

Джон фон Нейман

В основу більшості моделей сучасних комп'ютерів покладено архітектуру американського математика Джона фон Неймана — опис логічної організації ЕОМ (мал. 3.2).



Мал. 3.2

Основні складові фон-нейманівської машини такі: пристрій управління, арифметико-логічний пристрій, пам'ять, пристрої введення та виведення даних.

Принципи роботи комп'ютера за фон Нейманом

1. За допомогою пристроїв введення дані та програми їх опрацювання потрапляють у пам'ять комп'ютера.
2. З пам'яті комп'ютера дані надсилаються до процесора.
3. Арифметико-логічний пристрій здійснює опрацювання даних.
4. Пристрій управління забезпечує виконання процесів опрацювання даних, їх збереження та передавання.
5. Пристрої виведення даних здійснюють подання результатів опрацювання даних у вигляді, зручному для користувача.

3. Чи може комп'ютер працювати без процесора?

Процесор називають електронним «мозком» комп'ютера. Він призначений для автоматичного опрацювання й перетворення даних за наперед введеними програмами та управління роботою всіх пристроїв комп'ютера. Від його обчислювальної потужності здебільшого й залежить продуктивність комп'ютера.

Процесор — це мікросхема (мал. 3.3), яка створюється на напівпровідниковому кристалі (або кількох кристалах) шляхом застосування складної мікроелектронної технології. Різноманітні операції в процесорі виконуються за спеціальними вказівками. Вказівки для процесора записують у комп'ютерній програмі.

Під час роботи процесор досить сильно нагрівається, тому на нього встановлюють систему охолодження — вентилятор, який називають **кулером**.

Процесор складається з:

- арифметико-логічного пристрою для виконання арифметичних і логічних операцій із даними;
- пристрою управління для забезпечення функціонування всіх складових комп'ютера;
- регістрів власної пам'яті.

Під час роботи процесор опрацьовує дані. Частина даних інтерпретується як власне дані, частина даних — як адресні дані, а частина — як вказівки. Сукупність різноманітних вказівок, які може виконати процесор над даними, утворює так звану систему вказівок процесора.

Основними характеристиками процесорів є:

- **тип** — відповідно до фірми-виробника розрізняють процесори Intel (Pentium, Celeron, Core 2 Duo тощо), AMD (AMD64, Duron, Athlon тощо) та інші;
- **тактова частота** — визначає кількість виконуваних елементарних операцій за одну секунду, тобто швидкодію процесора; тактова частота сучасних процесорів вимірюється в гігагерцах (ГГц); уже розроблено процесори з частотою понад 3 ГГц;
- **розрядність** — максимальна довжина двійкового коду, який може опрацьовуватись або передаватися процесором; чим вищою є розрядність, тим потужніший процесор;
- **кеш-пам'ять** — це внутрішня пам'ять процесора, яка дає можливість зберігати проміжні дані.

Цікаво

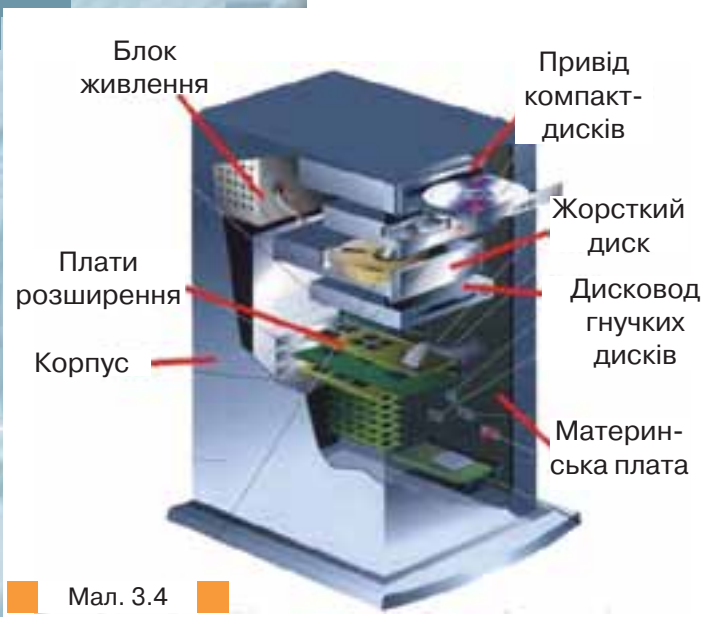
Принципи, згідно з якими функціонує більшість сучасних комп'ютерів, опубліковано в 1946 р. американським математиком Джоном фон Нейманом (1903–1957). Він також описав машину, яка може бути універсальним засобом опрацювання даних.

Мал. 3.3

Цікаво

Процесор ще іноді називають **CPU** — від англ. *Central Processing Unit* — модуль центрального процесора.

Кулер — від англ. *cooler* — охолоджувач.



Мал. 3.4

Режим роботи процесора задається спеціальною мікросхемою, яка називається генератором тактової частоти. Цей пристрій визначає ритм і швидкість роботи — на виконання процесором кожної операції відводиться певна кількість тактів.

Обсяг кеш-пам'яті сучасних процесорів коливається від 256 до 1024 Кб.

Розрядність пов'язана з розмірами спеціальних комірок пам'яті, що містяться в самому процесорі й називаються регістрами. Процесор із регістром 1 байт (8 біт) називають 8-розрядним, 2 байти — 16-розрядним, 4 байти — 32-розрядним. Найпотужніші комп'ютери сьогодні мають 8-байтові регістри (64 розряди).

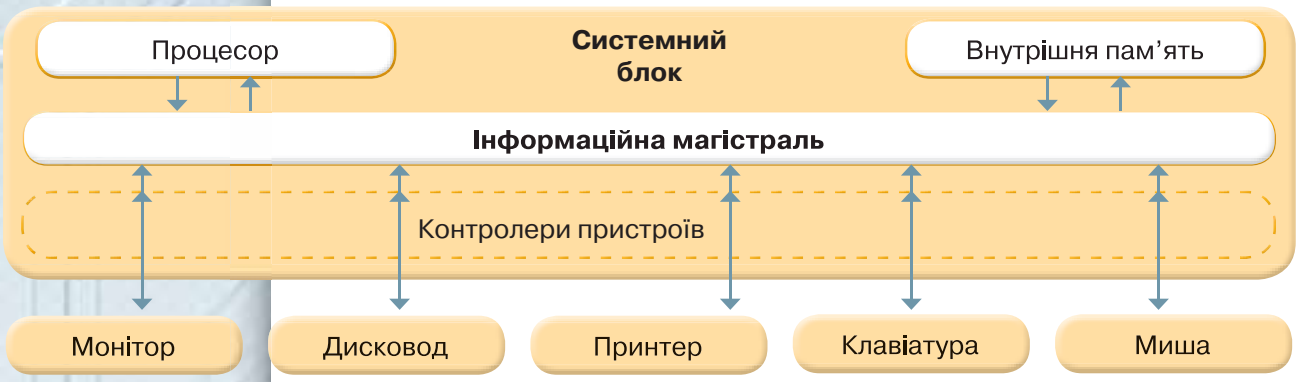
4. Які пристрої комп'ютера розміщуються всередині системного блока?

Деякі пристрої розміщені всередині системного блока комп'ютера (мал. 3.4), а інші приєднують до нього, тому вони належать до зовнішніх.

Усередині комп'ютера розміщується **системна плата**, яку ще називають **материнською**. На ній встановлені процесор, внутрішня пам'ять комп'ютера та інші пристрої. Процесор з'єднано з іншими пристроями пам'яті та пристроями для передавання даних і службових сигналів за допомогою набору електронних ліній, які називають **магістраллю (шиною)**. Користувач може створювати різні конфігурації комп'ютера, приєднуючи до магістралі окремі модулі різних пристроїв введення та виведення, пам'яті тощо (мал. 3.5).

Для магістралі характерна така організація: через одну групу проводів (шину даних) передаються дані, що опрацьовуються, через іншу (шину адрес) — адреси пам'яті або зовнішніх пристроїв, до яких «звертається» процесор. Через третю частину магістралі (шину управління)

Мал. 3.5



передаються управляючі сигнали (наприклад, перевірка готовності пристрою до роботи, сигнал до початку роботи пристрою тощо).

Користувач може змінювати набір пристроїв комп'ютера. Апаратне під'єднання зовнішніх пристроїв до магістралі здійснюється через **контролери** й **адаптери** — електронні мікросхеми, за допомогою яких узгоджують роботу зовнішніх пристроїв. Вони призначені для перетворення даних, що надходять із процесора, на відповідні сигнали, за допомогою яких здійснюється управління роботою пристрою. Їх роз'єми виведені на задній панелі системного блока й за допомогою відповідних кабелів до них приєднують зовнішні пристрої.

ДІЄМО

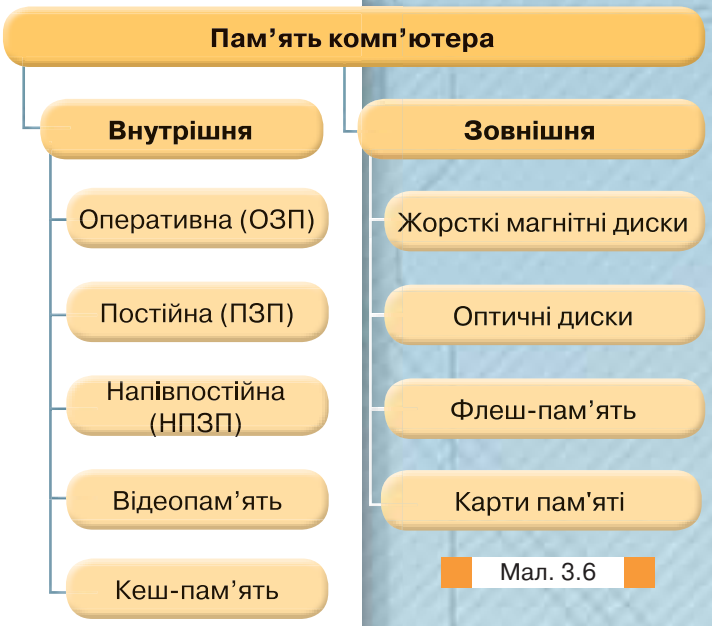
Вправа 2. Системний блок.
Завдання. В емуляторі системного блока розмістіть пристрої у правильному місці та правильному порядку.

1. Відкрийте програму *Системний блок*, збережену в папці *Забезпечення комп'ютера*.
2. Послідовно розміщуйте блок живлення, материнську плату та інші пристрої в місця їх правильного розташування.
3. Перевірте, чи отримано правильний результат.

5. Що і як можна зберігати в пам'яті комп'ютера?

Пам'ять комп'ютера призначена для зберігання даних і програм. Її поділяють на внутрішню та зовнішню (мал. 3.6).

Пристрої внутрішньої пам'яті виготовляють у вигляді мікросхем (модулів), які вставляються в спеціальні роз'єми на материнській платі.



Мал. 3.6

Внутрішня пам'ять комп'ютера поділяється на оперативний запам'ятовуючий пристрій (ОЗП), постійний запам'ятовуючий пристрій (ПЗП), напівпостійний програмований запам'ятовуючий пристрій (НПЗП), відеопам'ять і кеш-пам'ять.

Розглянемо пристрої внутрішньої пам'яті.

Оперативний запам'ятовуючий пристрій — ОЗП (RAM — від англ. *Random Access Memory* — пам'ять із довільним доступом) — швидка та енергозалежна пам'ять (мал. 3.7). Оперативна пам'ять призначена для тимчасового зберігання вхідних даних, проміжних і кінцевих результатів обчислень, програм опрацювання даних. Це своєрідний робочий простір для комп'ютера. ОЗП може використовуватися як для читання даних, так і для записування. Дані в ОЗП зберігаються доти, доки на їх місце не будуть записані нові дані. При вимкненні електроживлення дані в ОЗП втрачаються. Оперативна пам'ять сучасних комп'ютерів має обсяги 128, 256, 512, 1024 Мб і навіть сягає 4 Гб.

Мал. 3.7

Мал. 3.8

Постійний запам'ятовуючий пристрій — ПЗП (ROM — від англ. *Read Only Memory* — пам'ять тільки для читання) — швидка та енергонезалежна пам'ять (мал. 3.8). Дані заносяться до неї один раз назавжди (як правило, у заводських умовах) і зберігаються постійно (при ввімкненому й вимкненому живленні). Постійна пам'ять — мікросхема, в якій містяться програми для управління роботою комп'ютера та програми тестування основних складових комп'ютера, а також набір програм для управління всіма його пристроями (BIOS — від англ. *Basic Input/Output System* — базова система введення-виведення). Постійна пам'ять також розміщена на материнській платі.

Дані, що зберігаються в **напівпостійному програмованому запам'ятовуючому пристрої** — НПЗП (пам'ять, виконана за технологією CMOS — від англ. *Complementary Metal-Oxide Semicondactor* — технологія виготовлення мікросхем), можуть бути замінені у спеціальному режимі роботи комп'ютера — режимі програмування, коли користувач має спеціальні знання й може написати спеціальні програми для управління комп'ютером. До таких даних належать дані щодо зберігання і зміни конфігурації комп'ютера, календаря та годинника. НПЗП також називають пам'яттю автономного живлення, або пам'яттю «на батарейках», оскільки дані зберігаються за допомогою акумуляторної батарейки, за своїми функціями подібної до батарейок кварцевих годинників.

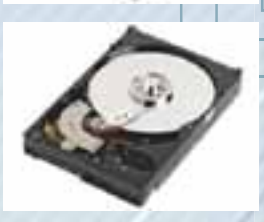
Мал. 3.9

Відеопам'ять — швидка оперативна пам'ять для зберігання коду зображення, що відображається на екрані монітора. Відеопам'ять (VRAM — від англ. *Video Random Access Memory*) розміщена на відеокарті (мал. 3.9). Найпродуктивніші відеокарти застосовують для комп'ютерних ігор або для роботи з просторовими зображеннями.

Чим більшою є ємність **відеопам'яті** комп'ютера, тим більші можливості відображення на моніторі графіків з високою роздільною здатністю й великою кількістю кольорів. Ємність відеопам'яті сучасних комп'ютерів становить 64, 128, 256 Мб і більше.

Кеш-пам'ять — це спеціальний вид пам'яті або частини ОЗП, де зберігаються копії часто використовуваних даних. Кеш-пам'ять забезпечує швидкий доступ до них.

Кеш-пам'ять сучасних комп'ютерів має кілька рівнів: кеш-пам'ять першого рівня ємністю 32 КБ вбудовується в процесор, а кеш-пам'ять другого рівня ємністю до 2 Мб зазвичай розміщується на материнській платі.



Мал. 3.10

Для тривалого зберігання даних призначено **зовнішню пам'ять**, або носії даних:

- жорсткий магнітний диск (ЖМД, HDD — від англ. *Hard Disk Drive*), або вінчестер (мал. 3.10). Як правило, вбудований разом із дисководом у корпус системного блока (він може бути розміщений і зовні);
- лазерні диски (CD-ROM, CD-R, CD-RW чи DVD) (мал. 3.11):
  - диски CD-ROM (від англ. *Compact Disk Read Only Memory* — компакт-диски тільки для читання) — високонадійні носії для зберігання даних, довговічні (термін придатності, що прогнозується при якісному виконанні, до 50 років). Діаметр диска може бути як 5,25, так і 3,5 дюйма. Принцип запису і зчитування — оптичний;
  - диски CD-R (від англ. *Compact Disc Recordable* — компакт-диск із одноразовим записуванням) — це різновид оптичного диска, на якому можна записати файли за допомогою записуючого пристрою.

    Записані дані можна прочитати з диска за допомогою приводу CD-ROM або відтворити у програвачі компакт-дисків (якщо це музика).

Мал. 3.11

На цей вид оптичних дисків можна записувати додаткові дані, аж доки закінчиться вільне місце. Зазвичай на диску CD-R можна помістити 650 Мб даних, або 74 хв музики. На CD-R дисках новішого покоління можна додатково збільшити цей ліміт до 737 Мб, або 80 хв музики. Сучасні CD-R можуть мати ємність до 800 Мб;

- диски CD-RW (від англ. *Compact Disc ReWritable* — компакт-диски з можливістю перезаписування) — це інший вид оптичних дисків, на якому можна не тільки записувати дані, а й вилучати й перезаписувати їх. Вони мають таку саму ємність, як CD-R;
- диски DVD (від англ. *Digital Video Disc* — цифровий відеодиск або *Digital Versatile Disc* — цифровий багатофункціональний диск) — це різновид носія даних, який зовні нагадує диск CD-ROM. Однак на DVD-диску можна записати значно більше даних.

Стандартна ємність цих носіїв становить 4,7 Гб, хоча трапляються диски удвічі більшої ємності. На DVD-диску можна записати з досконалою якістю повнометражний фільм у кількох мовних версіях;

- флеш-пам'ять, або USB-накопичувачі (від англ. *Universal Serial Bus* — універсальна послідовна шина) (мал. 3.12), під'єднуються безпосередньо до порту USB на комп'ютері. Вони являють собою мікросхеми й можуть зберігати до кількох гігабайтів даних;
- карти пам'яті, що використовуються для збереження даних у цифрових фотоапаратах, смартфонах та інших пристроях (мал. 3.13). Для перенесення на комп'ютер об'єктів, збережених на картах пам'яті, використовують кардрідер.

Мал. 3.12

Мал. 3.13

6. Які особливості мають пристрої введення та виведення даних?

Пристрої введення та виведення (мал. 3.14) призначені для введення даних до комп'ютера та виведення результатів їх опрацювання у вигляді, зручному для користувача.

Мал. 3.14

З основними пристроями введення даних — клавіатурою і мишею — ви вже ознайомилися раніше.

Основним пристроєм виведення даних є **монітор (дисплей)**. Монітори мають такі характеристики:

Мал. 3.15

Мал. 3.16, *а*

Мал. 3.16, *б*

- якість відображення кольорових зображень, тобто кількість кольорів для відображення;
- роздільна здатність, що визначається кількістю точок (пікселів) на екрані, які використовуються для створення зображення. Роздільна здатність подається як добуток кількості пікселів по горизонталі на кількість пікселів по вертикалі; наприклад, 800×600 або 1024×768. Чим вищою є роздільна здатність, тим детальніше можна відобразити зображення;
- довжина діагоналі в дюймах, розмір зерна (відстань на екрані між двома точками однакового кольору);
- максимальна частота відновлення зображення, на яку здатен монітор (її вимірюють у герцах), наприклад, 120 Гц. Чим вищою є частота відновлення, тим більша плавність відтворення зображення.

За принципом виготовлення розрізняють монітори на основі електронно-променевої трубки (мал. 3.15) та рідкокристалічні монітори (мал. 3.16, *а*, *б*).

Найдавнішим типом є монітори на основі електронно-променевої трубки — CRT (від англ. *Cathode Ray Tube* — катодно-променева трубка). CRT-монітор має дуже малі розміри зерна й незначне запізнення під час зміни зображення, він якісно відображає кольори. Проте такі монітори займають багато місця й у наш час активно витісняються рідкокристалічними моніторами.

У рідкокристалічних моніторів — LCD (від англ. *Liquid Crystal Display* — рідкокристалічний дисплей) — повністю відсутнє шкідливе електромагнітне випромінювання, використовується менше електроенергії, не створюється ефект мерехтіння, не спотворюється зображення. Вони мають плоский екран (мал. 3.16, *а*), або новітні моделі — вигнутий екран (мал. 3.16, *б*), і займають менше місця на столі. У рідкокристалічних моніторах застосовується технологія TFT (від англ. *Thin Film Transistor* — тонкоплівковий транзистор).

Для виведення зображень на монітор використовують відеопам'ять, що розташовується на відеокарті (мал. 3.9). Усе, що користувач бачить на екрані монітора, міститься у відеопам'яті, з якої надходять відеосигнали до монітора. **Відеокарта**, яку також називають **відеоадаптером**, установлюється всередині комп'ютера й використовується для підключення монітора та надсилання до нього графічних даних.

Параметри відеокарти визначають максимальні роздільну здатність, кількість кольорів і частоту відновлення зображення. Чим більшим є обсяг відеопам'яті, тим більшою кількістю пікселів на екрані може керувати відеоадаптер, тобто мати більш високу роздільну здатність екрана. Чим більший обсяг відеопам'яті використовується для керування одним пікселем, тим більшою є кількість відтворюваних кольорів, отже, багатшою — кольорова палітра монітора.

Сучасні комп'ютери мають монітори здебільшого з такими характеристиками:

- кількість кольорів — 256 і більше;
- роздільна здатність — 800×600 (1024×768, 1280×1024);
- розмір екрана — 15, 17, 19 дюймів.

Для виведення даних на папір, плівку чи інший носій використовують **принтери** (мал. 3.17). Основними характеристиками принтерів є якість друку (dpi — кількість точок на дюйм) та швидкість друку (кількість сторінок за хвилину — с./хв).

Цікаво

Зазвичай ноутбуки не оснащені відеоадаптерами. Вони мають інтегровану графіку або графічний процесор (GPU), вбудований у комп'ютер замість відеоадаптера.

Для домашнього використання зазвичай використовують **струменевий** принтер, який дає змогу друкувати як чорно-білі, так і кольорові документи. У струменевих принтерах для формування зображення використовуються спеціальні сопла, через які на папір подаються чорнила. Тонкі, як волосся, сопла містяться на друкувальній головці принтера, де встановлено резервуари з рідкими чорнилами, які, як мікрочастинки, переносяться через сопла на матеріал носія. Кількість сопел залежить від моделі принтера та його виробника. Зазвичай їх буває від 16 до 64. Деякі останні моделі мають набагато більшу кількість сопел: так, наприклад, головка принтера DeskJet 1600 має 300 сопел для чорних чорнил і 416 — для кольорових. Такі принтери здійснюють якісний малошумний друк. Але вони вимогливі до паперу, потребують дорогих витратних матеріалів, чорнило може розпливатись у воді.

Матричні принтери історично були найпершими принтерами і свого часу — найпоширенішими. Зараз ці принтери займають частку менше 10 %. Невисока якість чорно-білого друку до 300 dpi, можливість друку кількох копій одночасно під копіювальний папір та достатня дешевизна друку забезпечує їх використання в банках і для друку чеків. Працює матричний принтер шумно та з низькою швидкістю. Спосіб друку матричних принтерів схожий на спосіб друку звичайної друкарської машинки, але матричний принтер сам формує будь-яке зображення символів за допомогою голок, які б'ють по фарбувальній стрічці.

В офісах, де є потреба у друці великої кількості документів з невеликим шумом, використовують **лазерні** принтери. Якість друку на таких принтерах сягає до 1200 dpi, а швидкість — 25–50 с./хв. Витратні матеріали для них відносно недорогі, хоча є вимоги до якості паперу. Як і копіювальні апарати, лазерні принтери використовують фотобарабан, на який лазерним променем наноситься зображення. Електрично заряджений порошок — тонер — притягається до того місця, де відпрацював лазерний промінь. Тонер переноситься на папір і закріплюється на ньому гарячим спіканням за допомогою розігрітого валика.

Мал. 3.17

До **спеціальних** принтерів відносять фотопринтери, 3D-принтери та інші пристрої, які використовуються для друку професійних зображень чи об'єктів; у побуті, як правило, їх не використовують.

3D-принтер — пристрій, що працює методом пошарового створення фізичного об'єкта за цифровою 3D-моделлю (об'ємною). Існує декілька технологій 3D-друку від формування об'єкта з порошку до поступового його нашарування зі спеціального полімеру. Такі принтери використовують як у малосерійній інженерії, так й у виробництві складних систем, навіть біологічних. Ці технології останнім часом дуже швидко розвиваються, поступово стають дешевшими та більш доступними для користувачів.

Мал. 3.18

Мал. 3.19

7. Які пристрої входять до складу мультимедійного обладнання?

Використовувати мультимедійні програми та опрацьовувати мультимедійні дані можна лише за наявності в комп'ютері відповідного обладнання.

Мінімальний набір мультимедійного обладнання складається зі **звукової карти** (плати, що приєднується до материнської плати), до якої через відповідну панель системного блока під'єднується **акустична система (колонки)**, та **накопичувача для оптичних дисків**. Звук, який чує користувач комп'ютера, — результат роботи двох взаємопов'язаних компонентів: звукової карти (мал. 3.18) та акустичної системи. Їхній вибір залежить від потрібної якості звука та від сфери використання ПК (ігри, домашній мультимедійний центр, домашній кінотеатр для перегляду DVD-відео тощо). Проте якість відтворення звуку залежить не лише від пристроїв, а й від програмного забезпечення.

Важливими характеристиками акустичних систем є:

- діапазон частот. Зазвичай, у межах від 20 Гц до 20 кГц. Це доволі широкий діапазон, і для його відтворення потрібно кілька динаміків;
- кількість динаміків. Кожний динамік відтворює свій вузький діапазон частот;
- потужність. Становить від 2 до 180 Вт.

Замість колонок можна використовувати навушники (мал. 3.19). Роз'єм для навушників є в багатьох колонках або на задній панелі системного блока. Популярними на сьогодні є колонки та навушники з просторовим звуком.

До ширшого комплекту мультимедіа-системи належать **мікрофон**, **відеокамера**, **відеопрогравач**, **цифрова фотокамера** тощо.

З допомогою мікрофона можна записати звуковий фрагмент і зберегти його як файл. Зазвичай мікрофон використовують для спілкування в Інтернеті засобами IP-телефонії.

Мікрофон не є базовим мультимедійним пристроєм, тому, перш ніж його купити, варто впевнитись у сумісності мікрофона зі встановленою звуковою платою.

Мікрофон під'єднується до відповідного роз'єму звукової плати або до лінійного входу. У більшості мікрофонів є вимикач для вимикання вихідного сигналу (замість від'єднання від звукової плати).

Фізичне під'єднання зазначених пристроїв має супроводжуватися встановленням відповідних програм — **драйверів** (від англ. *to drive* — управляти, вести), які управляють роботою зовнішніх пристроїв комп'ютера. Як правило, відповідні драйвери розміщуються на CD-дисках і входять до комплекту під час продажу пристрою, оскільки кожний тип зовнішнього пристрою має індивідуальний драйвер. Створюючи мультимедійний центр, слід пам'ятати, що ефективна робота на комп'ютері з відео та графікою потребує особливих характеристик процесора, оперативної пам'яті, жорсткого диска.

8. Як обрати найкращий комп'ютер для роботи?

Якщо користувач не збирається працювати з професійними графічними програмами або опрацьовувати велику кількість даних, то йому не потрібна найдорожча модель і можна вибрати «повільніший» процесор.

Треба також зважати на те, що деякі складові комп'ютера впливають на швидкість перетворення даних, тобто поліпшують продуктивність комп'ютера (наприклад, пам'ять RAM, відеокарта), а інші — покра-

щують комфорт роботи користувача (ємність жорсткого диска, розмір і яскравість монітора, якість звукової карти).

Основні характеристики персонального комп'ютера залежать від характеристик його складових.

Але в будь-якому разі для здійснення зваженого вибору комп'ютера необхідно бути обізнаним щодо призначення, функціональності та особливостей роботи основних його складових.

ДІЄМО

Вправа 3. Конфігурація комп'ютера.

Завдання. Визначте характеристики комп'ютера, за яким ви працюєте.

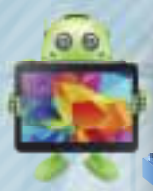
1. Клацніть правою кнопкою миші на значку *Комп'ютер* на *Робочому столі* й оберіть вказівку *Властивості*.
2. Перегляньте та проаналізуйте такі характеристики комп'ютера в області *Перегляд загальних відомостей про комп'ютер*: версія операційної системи, характеристики процесора, обсяг оперативної пам'яті, тип системи.

ОБГОВОРЮЄМО

Обговоріть запитання, що містяться у файлі *Тема 3*, який зберігається у папці *Обговорюємо*. Також відкрити ці запитання можна за допомогою QR-коду.

ПРАЦЮЄМО В ПАРАХ

1. Наведіть три приклади різних видів комп'ютерів, з якими ви стикаєтесь у повсякденному житті. Визначте їхні спільні ознаки. Результати подайте діаграмою Венна. Порівняйте отриманий результат із діаграмою іншої пари.
2. Обговоріть у парах: для чого призначені окремі пристрої комп'ютера. Один учень називає пристрій, що належить до комп'ютера, а інший — його призначення; потім поміняйтеся ролями та стежте, щоб назви пристроїв не повторювались.
3. Упорядкуйте носії даних за ступенем надійності, вартості, швидкості доступу, новизни тощо. Результати обговоріть у парах.
4. Без яких пристроїв не можна слухати музику на комп'ютері? За яких умов можна прослухати на комп'ютері концерт класичної музики, що записаний на CD?
5. За яких умов на комп'ютері можна переглянути відеофільм? А грати в комп'ютерну гру?
6. Чи тільки в комп'ютерах є процесори? Назвіть інші технічні пристрої, які можуть містити процесор. Де використовуються ці пристрої? Які функції виконують процесори в наведених вами прикладах? Чи схожі їх функції на функції процесора ПК? Чи можуть існувати ці пристрої без процесорів? Чи зміниться при цьому їх призначення та ефективність?
7. Знайдіть у довідковій літературі дані про сучасні процесори та перспективи їх розвитку. Якою є основна технологічна проблема створення сучасних процесорів? Чи виготовляє Україна процесори для комп'ютерів? Які корпорації є відомими виробниками процесорів? Обговоріть, як краще подати отримані відомості, щоб поділитися ними зі своїми однокласниками. Складіть план виступу.



ПРАЦЮЄМО САМОСТІЙНО

1. Створіть модель комп'ютера, скориставшись набором пристроїв, збережених у папці *Пристрої* папки *Забезпечення комп'ютерів* для:
   а) навчання та досліджень;
   б) ігор та розваг;
   в) ведення ділової документації.
2. Знайдіть в Інтернеті відеозаписи та зображення різного приладдя для лічби, яке використовувалося в давнину.
3. Знайдіть в Інтернеті, у які роки змінювались етапи розвитку засобів для обчислення, що відображені на малюнку 3.1.
4. Сплануйте та складіть презентацію *Україна в історії обчислювальної техніки*.
5. Складіть карту знань класифікації пристроїв комп'ютера. Передбачте в ній стандартні та додаткові пристрої введення, виведення, запам'ятовування та процесор. Скористайтеся для довідки файлом *Додаткові пристрої,* який збережений у папці *Забезпечення комп'ютерів.*
6. Визначте конфігурацію домашнього комп'ютера. Скористайтеся цінами в інтернет-магазині та визначте теперішню вартість подібного комп'ютера. Зробіть висновок.

4. ПРАКТИЧНА РОБОТА 2

КОНФІГУРАЦІЯ КОМП'ЮТЕРА ПІД ПОТРЕБУ КОРИСТУВАЧА

ПРИГАДАЙТЕ

- призначення складових комп'ютера;
- типову архітектуру персонального комп'ютера;
- основні характеристики запам'ятовуючих пристроїв, процесорів, моніторів, відеоадаптерів, принтерів.

СТВОРІТЬ

У власній структурі папок створіть папку *Практична робота 2.*

ПАМ'ЯТАЙТЕ

Під час виконання практичних завдань пам'ятайте про правила безпеки життєдіяльності при роботі з комп'ютером!

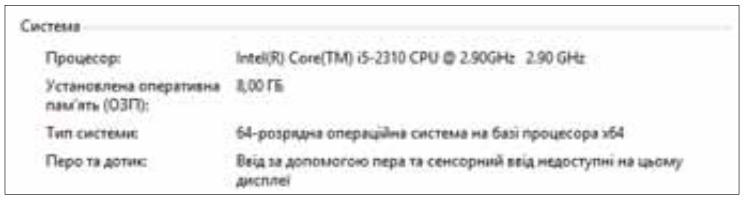
Завдання 1. Конфігурація комп'ютера (*12 балів*)

Запропонуйте, як обладнати комп'ютером робоче місце для:

- учня на олімпіаді з комп'ютерної графіки;
- співробітника бюджетної установи, який працюватиме в бухгалтерії;
- учня, який буде використовувати мультимедіа та ігри;
- студента навчального закладу;
- офісного працівника, який працюватиме з великим обсягом документів, затративши мінімум ресурсів.

Призначення комп'ютера для обладнання оберіть із запропонованого списку. Наведіть два приклади конфігурації. Укажіть загальну вартість обладнання. Результати роботи подайте в презентації.

Завдання 2. Призначення комп'ютера (*5 балів*)
За конфігурацією комп'ютера визначте, які завдання можна виконувати на ньому. Для яких користувачів ви б його порадили? Запишіть свої висновки до тексту електронного листа та надішліть його на навчальну електронну скриньку, яку вкаже вчитель.


Система

Процесор:	Intel(R) Core(TM) i5-2310 CPU @ 2.90GHz 2.90 GHz
Установлена оперативна пам'ять (ОЗП):	8,00 ГБ
Тип системи:	64-розрядна операційна система на базі процесора x64
Перо та дотик:	Ввід за допомогою пера та сенсорний ввід недоступні на цьому дисплеї

Завдання 3. Пристрої введення та виведення (*12 балів*)
Створіть текстовий документ із рекомендаціями щодо придбання пристроїв введення та виведення даних. У документі передбачте назву пристрою, 2–3 приклади моделей і їх зображення, ціну та посилання на інтернет-магазин для можливого придбання, поради щодо завдань, які можна виконувати за допомогою цього пристрою. Необхідні відомості знайдіть в Інтернеті.

5. ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ КОМП'ЮТЕРА

ПРИГАДАЙТЕ:

- що називається програмою; де відображаються назви програм, установлених на комп'ютері;
- як запустити програму на виконання;
- які програми називають операційною системою;
- коли слід дотримуватись авторського права;
- що таке інтерфейс операційної системи та як він реалізується в операційній системі *Windows 7*.

ВИ ДІЗНАЄТЕСЯ:

- яке програмне забезпечення потрібне для роботи комп'ютера;
- якими є правила використання програмного забезпечення;
- які існують види ліцензій на програмне забезпечення;
- для чого потрібна операційна система, які її функції;
- як класифікують операційні системи;
- які програми належать до службових програмних засобів;
- коли програмне забезпечення потребує інсталяції та деінсталяції;
- у яких випадках говорять про проблеми сумісності програмного забезпечення;
- що таке форматування носіїв даних і за допомогою якого програмного забезпечення воно здійснюється;
- які функції виконують програми-архіватори;
- у чому полягають основні методи стиснення даних.

ВИВЧАЄМО

1. Яке програмне забезпечення потрібне для роботи комп'ютера?

Ви вже знаєте, що без програм комп'ютер не зможе працювати, а буде лише набором електронних пристроїв.

Мал. 5.1

Для розв'язування задач на комп'ютері потрібно, щоб кожна програма була налагоджена та мала відповідну документацію. Тому стосовно роботи на комп'ютері часто використовують термін **програмне забезпечення (software)**, під яким розуміють сукупність програм і правил, а також документації, щодо функціонування комп'ютера для опрацювання даних.

Розрізняють програми різного рівня, кожний з яких має відповідне призначення. Схематично структуру програмного забезпечення наведено на малюнку 5.1.

Програми **базового рівня** зберігаються у спеціальних мікросхемах постійного запам'ятовуючого пристрою та утворюють базову систему введення-виведення — BIOS. Програми та дані записуються в ПЗП на етапі виробництва й не можуть бути змінені в процесі експлуатації. Програми цього рівня забезпечують взаємодію з базовими апаратними засобами.

Програми **системного рівня** забезпечують взаємодію інших програм комп'ютера з програмами базового рівня та безпосередньо з апаратним забезпеченням. При під'єднанні до комп'ютера нового обладнання на системному рівні має бути встановлена програма, що забезпечує для решти програм взаємозв'язок із цим пристроєм. Конкретні програми, призначені для взаємодії з конкретними пристроями, називають **драйверами**. Інший клас програм системного рівня відповідає за взаємодію з користувачем. Завдяки таким програмам є можливість вводити дані до комп'ютера, управляти його роботою й отримувати результат у зручному для користувача вигляді. Це засоби забезпечення користувацького інтерфейсу, від них залежить зручність і продуктивність роботи з комп'ютером.

Програми **службового рівня** взаємодіють як із програмами базового рівня, так і з програмами системного рівня. Призначення службових програм (утиліт) полягає в автоматизації перевірки та налаштування комп'ютера, а також для покращення функцій системних програм, підвищення ефективності роботи комп'ютера та розширення можливостей його використання. До таких програм належать програми для роботи з архівами даних (наприклад, *7-zip*, *WinRar*), оптимізації розміщення даних на диску (наприклад, *defrag*), антивірусні програми (наприклад, *Intel Security-McAfee*, *Symantec Norton™ Security*, *Zillya! Internet Security*), програми тестування комп'ютера (наприклад, *ScanDisk*) тощо.

Програмне забезпечення **прикладного рівня** являє собою комплекс прикладних програм, за допомогою яких виконуються конкретні завдання (від виробничих до творчих, розважальних і навчальних). Між прикладним і системним програмним забезпеченням існує тісний взаємозв'язок. Прикладні програми призначені для комп'ютерної підтримки виконання прикладних завдань. Розрізняють прикладні програми загального та професійного (спеціального) призначення (мал. 5.2).

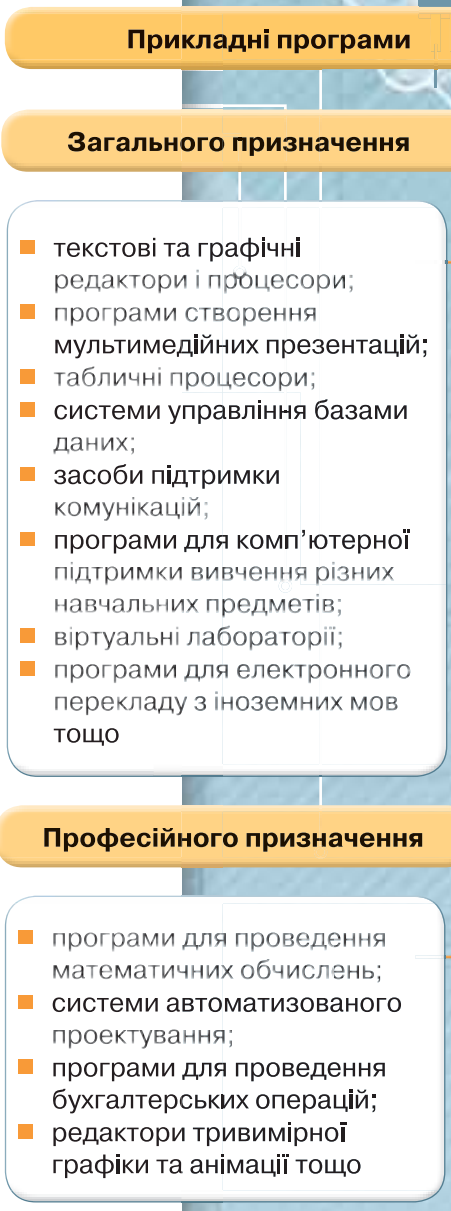
До прикладного програмного забезпечення **загального призначення** належать: текстові та графічні редактори і процесори, програми створення мультимедійних презентацій, табличні процесори, системи управління базами даних, засоби підтримки комунікацій тощо.

До прикладного програмного забезпечення загального призначення відносять також програми для комп'ютерної підтримки вивчення різних

навчальних предметів, іноземних мов, віртуальні фізичні та хімічні лабораторії, програми для електронного перекладу з іноземних мов, швидкого набору тексту на клавіатурі тощо. Досить зручно користуватись електронними довідниками та енциклопедіями. Головною їх перевагою перед паперовими аналогами є компактність і зручність у пошуку відомостей.

Прикладне програмне забезпечення **професійного (спеціального) призначення** потрібне для вузькоспеціалізованого використання. У таких програмах враховується специфіка конкретних задач, і складені вони на основі спеціальних методів подання та опрацювання даних, властивих конкретній галузі практичної діяльності людей. До програм професійного призначення належать програми для проведення математичних обчислень (наприклад, *Mathlab*, *MathCad*), системи автоматизованого проектування (наприклад, *AutoCad*), програми для проведення бухгалтерських операцій (наприклад, *1С-бухгалтерія*), редактори тривимірної графіки та анімації (наприклад, *3D MAX Studio*) тощо.

Для розробки як прикладного, так і системного програмного забезпечення комп'ютера існують спеціальні програми — **інструментальне програмне забезпечення**. Такими інструментальними засобами є **системи програмування**, призначені для сприймання та опрацювання програм, записаних однією чи декількома мовами програмування (наприклад, *C*, *C#*, *C++*, *Pyton, Visual Basic*, *Delphi*). Крім того, є системи, що підтримують кілька мов програмування, наприклад *Microsoft Visual Studio.NET*.



Мал. 5.2

2. Якими є правила використання програмного забезпечення?

Комп'ютерні програми створюють програмісти. Розробка нового програмного забезпечення (ПЗ) — трудомісткий і тривалий процес, що потребує глибоких знань і певних навичок, насамперед у галузі математики та інформатики. Будь-які програми мають ціну та власника, тобто комусь належать.

Більшість програмних продуктів є комерційними, тобто передбачають плату за їх використання. Наприклад, на платній основі розповсюджується таке програмне забезпечення, як операційна система *Windows*, пакет прикладних офісних програм *Microsoft Office*, антивірусне програмне забезпечення, програми для комп'ютерної підтримки навчання, комп'ютерні ігри.

Будь-яке програмне забезпечення розповсюджується на основі ліцензійних угод, а не лише продається та купується. Програми, що розповсюджуються безкоштовно, також передбачають ознайомлення з ліцензійною угодою та підтвердження користувачем дотримання правил використання програмного забезпечення, визначених такою угодою.

Ліцензія на програмне забезпечення — правовий документ, що визначає правила використання та поширення програмного забезпечення.

Мал. 5.3

Програмне забезпечення є об'єктом інтелектуальної власності, всі права на нього належать розробнику.

Це право захищається Законом України «Про авторське право та суміжні права». За цим законом під час продажу програмного забезпечення розробник не передає кінцевому користувачеві свої права на певну програму, а лише дозволяє використовувати (ліцензує) цю програму. На упаковці програмного продукту, як правило, зазначається стисла ліцензійна угода, яка визначає основні права та обов'язки виробника і власника одержаного програмного продукту.

На жаль, не всі програмні продукти, що продаються, є легальними. З юридичної точки зору до піратського програмного забезпечення належать усі комп'ютерні програми, які розповсюджуються, встановлюються на комп'ютери й використовуються з порушенням умов їх ліцензійної угоди. Наприклад, самостійно створені копії ліцензійної програми перестають бути легальними, а подібні дії є порушенням авторських прав і спричиняють правову відповідальність.

Ліцензійне програмне забезпечення здебільшого може записуватись на окремий компакт-диск і постачатись разом із супроводжувальною документацією в кольоровій картонній коробці або в спеціальній упаковці (мал. 5.3). Доступ до ліцензійного програмного забезпечення можна отримати також на сайті виробника, вказавши персональний пароль ліцензії.

3. Які існують види ліцензій на програмне забезпечення?

Розрізняють різні види ліцензій на програми. Основні з них (мал. 5.4):

- власницькі (пропрієтарні);
- вільні;
- відкриті.

Мал. 5.4

Вони істотно різняться щодо прав кінцевого користувача на використання програми.

Власницька, або **пропрієтарна, ліцензія** (від англ. *proprietary* — власницький) передбачає, що розробник ПЗ дає дозвіл користувачу використовувати одну або декілька копій програми, але при цьому сам залишається правовласником усіх цих копій. Таким чином, практично всі права на ПЗ залишено за розробником, а користувач здобуває лише дуже обмежений набір окреслених прав. Для пропрієтарних ліцензій типовим є накладання великої кількості умов, що забороняють певні варіанти використання ПЗ, навіть тих, які без цієї заборони були б дозволені законом про авторське право. Прикладом пропрієтарної ліцензії може бути ліцензія на операційну систему *Microsoft Windows*, яка включає великий список заборонених варіантів використання.

Найзначнішим наслідком застосування пропрієтарної ліцензії є те, що кінцевий користувач зобов'язаний прийняти її, бо за законом власником ПЗ є не користувач, а розробник програми. У разі відмови прийняти ліцензію користувач взагалі не може працювати з програмою.

Вільні та **відкриті ліцензії** не залишають права на конкретну копію програми її розробнику, а передають найважливіші з них кінцевому користувачу, який і стає власником. У результаті користувач отримує важливі права, які Закон про авторське право зазвичай дає лише власнику копії. Проте всі авторські права на ПЗ, як і раніше, залишаються в розробника.

Ліцензія *Freeware* (безкоштовне ПЗ) не потребує виплат правовласнику, не має обмежень щодо функціональності й часу роботи. Проте таке ПЗ

можна поширювати без тексту програми, і можуть бути обмеження щодо комерційного використання чи модифікації програми.

Ліцензія *Free software* (вільне ПЗ) надає максимальну кількість прав користувачам: користування, поширення, модифікація. Для цього типу ПЗ створюються спеціальні ліцензії для врегулювання прав та обов'язків авторів і користувачів.

Відмінність між безкоштовним та вільним ПЗ полягає в тому, що згідно з ліцензією *Freeware* користувачі не мають права поширювати програми, дарувати, модифікувати тощо, а згідно з *Free software* — це дозволяється. Часто, хоча не завжди, вони відрізняються ще й тим, що вільне ПЗ надають разом із текстом програми.

4. Для чого потрібна операційна система, які її функції?

Основою системного програмного забезпечення є операційна система (наприклад, *Windows*, *Linux*).

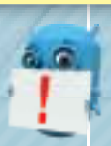
Операційна система (ОС) — це програмний комплекс, що забезпечує:
- управління ресурсами — злагоджену роботу всіх апаратних засобів комп'ютера;
- управління процесами — виконання всіх програм та їх взаємодію з пристроями комп'ютера та даними;
- взаємодію (обмін відомостями та даними) між користувачем і комп'ютером.

Операційна система починає працювати одразу після ввімкнення комп'ютера. Певну її частину — BIOS — розміщено на мікросхемі постійної пам'яті. Ця частина містить сукупність програм, які після ввімкнення комп'ютера автоматично тестують усі його пристрої та в разі їх справної роботи завантажують в оперативну пам'ять частину операційної системи — програму-завантажувач. Далі вже вона завантажує в оперативну пам'ять комп'ютера потрібні для подальшої роботи модулі операційної системи. Після завершення завантаження ОС управління переходить до командного процесора — частини ОС, що забезпечує виконання команд користувача. Поки комп'ютер працюватиме, деяка частина операційної системи завжди залишатиметься в ОЗП. Цю частину ОС називають **резидентною**. За потреби до оперативної пам'яті буде завантажено інші частини ОС.

До складу сучасних операційних систем входять такі основні компоненти (мал. 5.5):
- **ядро** — центральна частина ОС, що забезпечує прикладним програмам координований доступ до ресурсів комп'ютера (часу, що витрачається процесором для опрацювання окремих завдань, оперативної пам'яті, зовнішніх пристроїв уведення та виведення даних), перекладаючи їх команди з мови прикладних програм мовою двійкових кодів для подальшого опрацювання комп'ютером;
- **драйвери** — програми для перекладу вказівок комп'ютера мовою певного пристрою (принтера, сканера, звукової або відеокарти тощо) та навпаки;
- **утиліти** — допоміжні програми, призначені для обслуговування дисків, перевірки комп'ютера, налаштування параметрів роботи;
- **інтерфейс** — правила взаємодії операційної системи та користувача, які визначають зручність роботи.

Цікаво

Текст програми (вихідний код, програмний код, джерельний код, початковий код, сирцевий код, англ. *source code*) — набір інструкцій або вказівок, написаних комп'ютерною мовою програмування й у формі, що її може прочитати й модифікувати людина. Текст програми надає можливість програмісту вивчати та змінювати роботу програми залежно від потреб користувача.



Мал. 5.5

До основних функцій операційної системи належать:

- створення середовища виконання та взаємодії прикладних програм;
- розподіл апаратних ресурсів комп'ютера між прикладними програмами;
- надання прикладним програмам засобів для ефективного використання пристроїв і виконання типових операцій введення та виведення даних;
- зберігання даних пристроями пам'яті;
- надання інтерфейсу, за допомогою якого користувачі управлятимуть виконанням прикладних програм та вмістом пристроїв пам'яті;
- забезпечення взаємодії комп'ютерів у мережах.

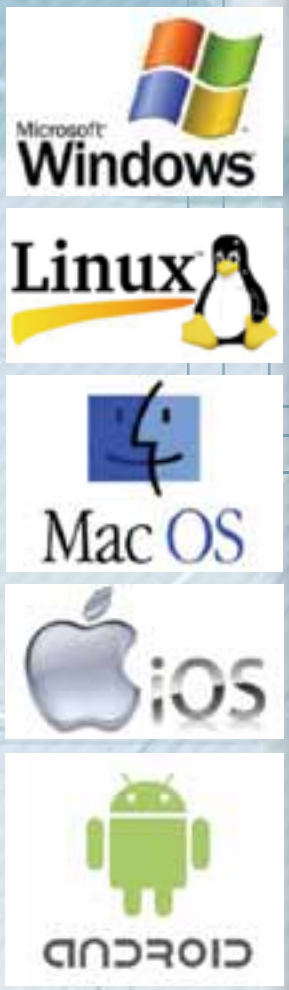
Останнім часом операційні системи на комп'ютери встановлюють виробники чи компанії, що займаються комплектацією, продажем та обслуговуванням комп'ютерів. Однак користувачі за потреби можуть самостійно встановити чи замінити операційну систему на своєму комп'ютері. Для цього необхідно придбати відповідний пакет програм, перевірити наявність ліцензії, оскільки комп'ютерні програми захищаються Законом про авторське право, та встановити за певними правилами операційну систему на комп'ютер.

5. Як класифікують операційні системи?

Операційні системи (ОС) можна класифікувати за такими ознаками:

- цільове призначення для:
  - великих універсальних високопродуктивних ЕОМ (мейнфреймів);
  - ПК;
  - мобільних пристроїв;
  - вбудованих систем;
- кількість користувачів, які одночасно працюють із системою: локальні — **однокористувацькі** та мережеві — **багатокористувацькі**;
- кількість задач, які може розв'язувати користувач за їх допомогою одночасно: **однозадачні** й **багатозадачні**;
- інтерфейс користувача — основний спосіб взаємодії користувача з ОС:
  - інтерфейс командного рядка — управління за допомогою введених із клавіатури команд;
  - графічний інтерфейс — вибір із меню або вказування на графічні зображення (WIMP-інтерфейс, від англ. *Window*, *Icon*, *Menu*, *Pointer* — вікно, значок, меню, вказівник);
  - SILK-інтерфейс — введення вказівок за допомогою голосу (від англ. *Speech*, *Image*, *Language*, *Knowledge* — мовлення, образ, мова, знання);
  - жестовий інтерфейс — управління за допомогою сенсорного екрана, джойстика тощо;
- ресурси, мінімально необхідні для її роботи: мінімальна ємність оперативної та дискової пам'яті, тип процесора;
- відкритість: можливість користувача, що знає мови програмування, вносити потрібні зміни в її окремі модулі;
- кількість розрядів, що опрацьовують дані у процесорі одночасно:
  - 16-розрядні;
  - 32-розрядні;
  - 64-розрядні.

Найвідомішими є операційні системи: *MS Windows*, *GNU/Linux*, *UNIX*, *OS/2*, *MacOS*, *iOS*, *Android* (мал. 5.6). Однією з перших відомих операцій-



Мал. 5.6

них систем була *MS-DOS*, яку в минулому встановлювали на більшість комп'ютерів, проте з часом її замінили на ОС із графічним інтерфейсом.

Наприклад, операційні системи *Windows* та *Linux* забезпечують роботу кількох користувачів одночасно (мережеві), а *MS-DOS* є однокористувацькою операційною системою; операційні системи *MS-DOS* та *UNIX* передбачають введення користувачем усіх вказівок із клавіатури, а при роботі з операційною системою *Windows* користувачеві, щоб вказати на виконання операції, достатньо вибирати на екрані комп'ютера графічні об'єкти та меню. Операційна система *Linux*, на відміну від ОС *Windows*, має відкритий код.

6. Які програми належать до службових програмних засобів?

До службових програм належать такі групи програм (мал. 5.7).

Диспетчери файлів (файлові менеджери). За їх допомогою виконується більшість операцій з обслуговування файлової структури: копіювання, переміщення, перейменування файлів, створення папок, видалення об'єктів, пошук файлів та навігація у файловій структурі. Базові програмні засоби містяться у складі програм системного рівня і встановлюються разом з операційною системою.

Архіватори (засоби стиснення даних). Призначені для створення архівів. Файли архівів мають підвищену щільність запису даних, за рахунок чого ефективніше використовуються носії даних.

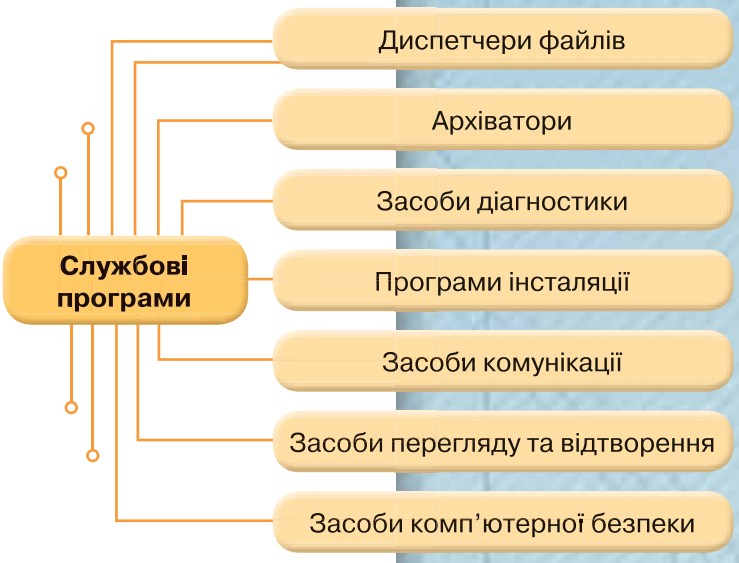
Засоби діагностики. Призначені для автоматизації процесів діагностування програмного та апаратного забезпечення. Їх використовують для виправлення помилок і для оптимізації роботи комп'ютерної системи.

Програми інсталяції (встановлення). Призначені для контролю за додаванням у поточну програмну конфігурацію нового програмного забезпечення. Вони слідкують за станом і зміною навколишнього програмного середовища, відслідковують і протоколюють утворення нових зв'язків. Прості засоби управління встановленням та вилученням програм містяться у складі операційної системи, але можуть використовуватись і додаткові службові програми.

Засоби комунікації. Дають змогу встановлювати з'єднання з віддаленими комп'ютерами, передають повідомлення електронної пошти, пересилають факсимільні повідомлення тощо.

Засоби перегляду та відтворення. Призначені для перегляду зображень та відтворення аудіо- або відеофайлів. Використовуються, якщо до файлів не потрібно вносити зміни.

Засоби комп'ютерної безпеки. До них належать засоби пасивного та активного захисту даних від пошкодження, несанкціонованого доступу, перегляду та зміни даних. Засоби пасивного захисту — це службові програми, призначені для резервного копіювання. Як засіб активного захисту застосовується антивірусне програмне забезпечення.



Мал. 5.7

Цікаво

На одному комп'ютері можна встановити декілька операційних систем. Для цього використовують **віртуальну машину** — спеціальну програму, яка забезпечує повну емуляцію фізичної машини чи середовища.

Для захисту даних від несанкціонованого доступу, їх перегляду та зміни використовують спеціальні системи, базовані на криптографії.

7. Коли програмне забезпечення потребує інсталяції та деінсталяції?

Більшість програм постачають для продажу та поширення у стисненому вигляді. Для нормальної роботи їх потрібно розпакувати, а необхідні дані — правильно розташувати на комп'ютері, враховуючи відмінності між комп'ютерами та налаштуваннями користувача. У процесі встановлення виконують різні тести на відповідність заданим вимогам, а комп'ютер необхідним чином налаштовують для зберігання файлів і даних, необхідних для правильної роботи програми.

Процес встановлення (інсталяція) програмного забезпечення на комп'ютер користувача може бути здійснений за допомогою:

- менеджера пакетів — особливої програми у складі операційної системи (наприклад, *APT* в *Linux*, *Програми та засоби* в *Microsoft Windows 7*);
- засобу встановлення — спеціальної програми у складі самого програмного забезпечення.

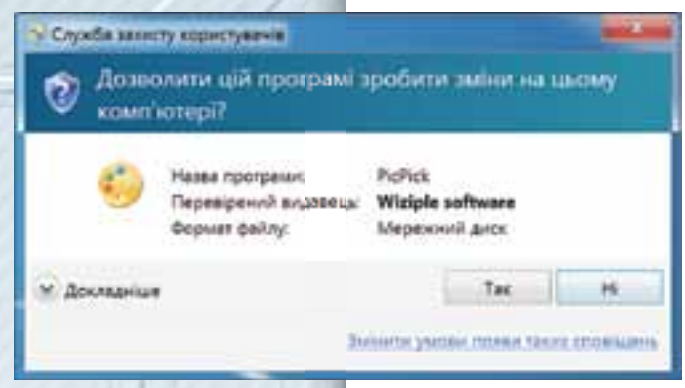
Встановлення програмного забезпечення зазвичай включає в себе розташування всіх необхідних програмі файлів у відповідних місцях файлової системи, а також зміну та створення конфігураційних файлів. Менеджери пакетів також виконують контроль залежностей, перевіряючи, чи є в системі необхідні для роботи даної програми засоби, а в разі успішного встановлення реєструють новий програмний засіб у переліку наявних.

Деякі комп'ютерні програми створені таким чином, що їх встановлюють простим копіюванням файлів у потрібне місце. Про такі програми кажуть, що вони не вимагають інсталяції, їх поширюють копіюванням.

Інсталяція великих програм буває:

- типова (англ. — *typical*, *normal*);
- мінімальна (англ. — *minimum*);
- повна (англ. — *full*);
- керована користувачем (англ. — *custom*).

Вилучення (деінсталяцію) програм потрібно здійснювати за допомогою системних чи спеціальних програмних засобів. Просте вилучення файлів не приводить до вилучення програми з реєстру встановленого програмного забезпечення.

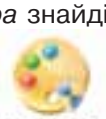


Мал. 5.8

ДІЄМО

Вправа 1. Інсталяція програми *PicPick*.
Завдання. Інсталюйте графічний редактор *PicPick*.

1. У папці *Забезпечення комп'ютера* знайдіть інсталяційний пакет *picpick_inst.exe*.



2. У вікні *Служба захисту користувачів* (мал. 5.8) підтвердіть намір встановити програму на своєму комп'ютері.

3. У вікні встановлення програми прочитайте ліцензійну угоду щодо використання програми. У разі потреби скористайтеся онлайновим перекладачем. Натисніть кнопку *I Agree* (мал. 5.9) — підтвердження того, що ви приймаєте умови угоди та продовжуєте інсталяцію програми.
4. Оберіть місце розташування програми: *C:\\Program Files(x86)\PicPick* та натисніть кнопку *Install.*
5. Дочекайтеся завершення процесу інсталяції.

Мал. 5.9

8. У яких випадках говорять про проблеми сумісності програмного забезпечення?

Сьогодні існує багато різних виробників як електронної складової, так і програмного забезпечення комп'ютера. Це призвело до проблем їх сумісності — здатності різних об'єктів взаємодіяти між собою. Якщо проблему апаратної сумісності вже майже вирішено, то проблема сумісності програмного забезпечення досі ще є актуальною. Розрізняють види сумісності програмного забезпечення на рівні:

- виконуваних файлів;
- програмних кодів (програма може бути виконана на різних комп'ютерах під управлінням різних операційних систем);
- форматів файлів даних (програми можуть відрізнятися за інтерфейсом, набором функцій, але працювати з однаковими документами);
- мережевої сумісності (здатність програм обмінюватися даними через мережу).

9. Що таке форматування носіїв даних і за допомогою якого програмного забезпечення воно здійснюється?

До службового програмного забезпечення належать також програми для обслуговування носіїв даних. Найбільш уживаною серед них є програма форматування. Її зазвичай виконують перед встановленням усіх програм на жорсткий диск, а також після придбання додаткового носія даних.

Форматування (англ. *formatting*) — процедура створення структур порожньої файлової системи вказаного типу — розподіл доріжок жорсткого магнітного диска чи іншого носія даних (наприклад, флеш-накопичувача чи карти пам'яті) на фізичні чи логічні записи, що виконується перед першим використанням диска. Форматування накопичувача, що містить дані, супроводжується втратою всіх даних.

У процесі форматування також можуть перевірятися цілісність носія й виправлятися пошкодження.

Пристрої зберігання даних, зокрема флеш-накопичувачі й карти пам'яті, можуть надходити в продаж уже відформатованими виробником.

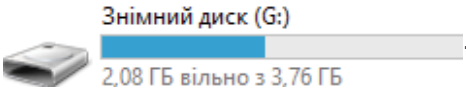
ДІЄМО

Вправа 2. Форматування флеш-накопичувача.

Завдання. Відформатуйте флеш-накопичувач.

Мал. 5.10

1. Приєднайте флеш-накопичувач до USB-роз'єма. Перевірте, чи відображається назва приєднаного пристрою у списку пристроїв вікна *Мій комп'ютер*.
2. Викличте контекстне меню знімного диска



. Оберіть вказівку *Форматувати*.
3. У списку доступних файлових систем оберіть *FAT32*. Задайте швидкий спосіб форматування — очищення змісту (мал. 5.10).
4. Натисніть кнопку *Почати*. Дочекайтеся, поки процес форматування не завершиться. Він буде супроводжуватись індикатором процесу виконання завдання в нижній частині вікна. Завершіть форматування, натиснувши кнопку *Закрити*.
5. Перевірте, чи було видалено вміст флеш-накопичувача в процесі форматування.

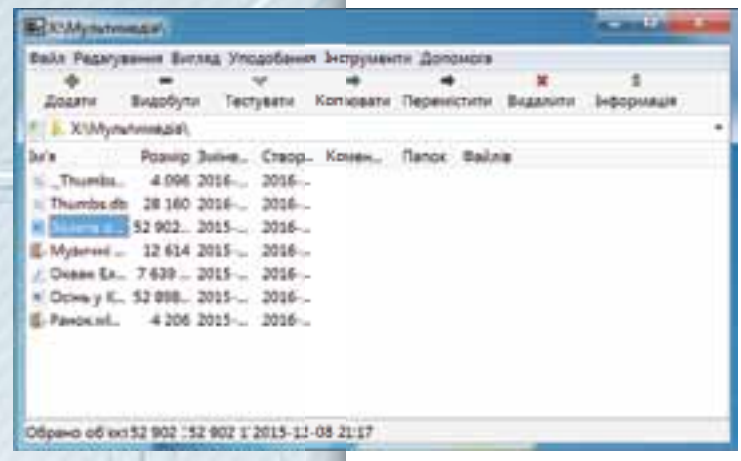
10. Які функції виконують програми-архіватори?

При передаванні даних через комп'ютерну мережу, а також при збереженні резервних копій файлів суттєвим є їх обсяг. Тому часто застосовують стиснення файлів. Стискати можна не лише один файл, а й папку, що містить кілька файлів чи папок. Результатом стиснення є запакований файл, або архів. Створювати такі файли та працювати з ними дають можливість спеціальні програми, які називають архіваторами і програмами резервного копіювання.

Часто вживаними є програми-архіватори *7-Zip* (мал. 5.11), *WinRar*, *WinZip*. Архіви, створені за допомогою цих програм, мають розширення відповідно *7z*, *rar*, *zip*.

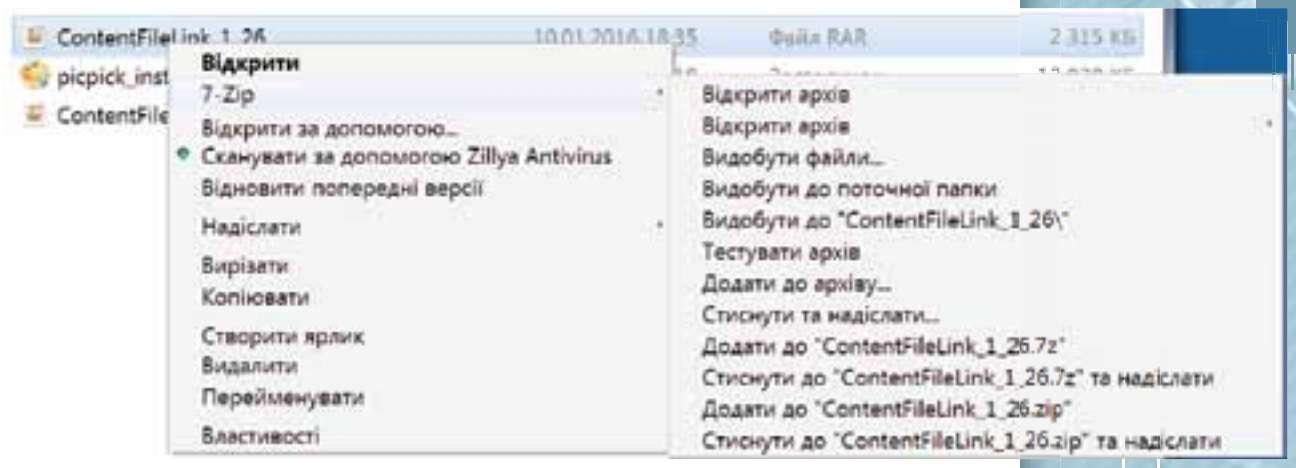
До базових функцій, які виконують більшість сучасних архіваторів, відносять:

- створення нових архівів;
- розпакування файлів з архівів (розархівування);
- додавання файлів до архіву;
- створення архівів, що саморозпаковуються;
- створення розподілених архівів на носіях малої ємності;
- тестування цілісності структури архівів;
- повне або часткове відновлення пошкоджених архівів;
- захист архівів від перегляду й несанкціонованої модифікації.

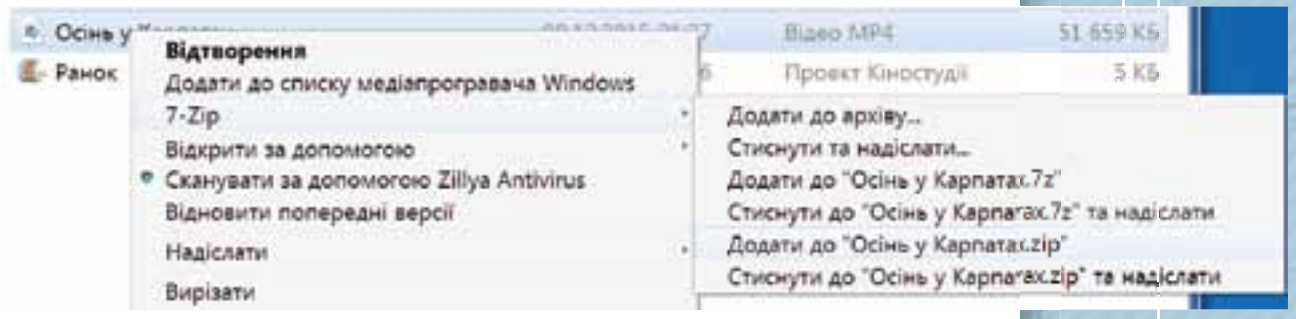


Мал. 5.11

Архівація передбачає упакування та стиснення даних. Упакування й стиснення (компресія) — не одне й те саме. Упакування — це злиття кількох файлів або папок у єдиний



Мал. 5.12



Мал. 5.13

файл, який називається архівом. Стиснення ж — скорочення обсягу вихідного файла або групи файлів.

У різних архіваторах застосовують різні способи стиснення, тому обсяг файла архіву порівняно з вихідним файлом може різнитися залежно від програми-архіватора, за допомогою якої його було створено.

Швидко створити файл архіву із значеннями його властивостей, що встановлені за замовчуванням, або розпакувати архів можна за допомогою вказівок контекстного меню. Якщо на комп'ютері встановлено програму-архіватор, то вказівки для виконання найбільш вживаних операцій з архівами виносяться в контекстне меню (мал. 5.15, 5.13).

Для додавання файлів до щойно створеного або відкритого архіву слід вибрати в програмі-архіваторі вказівку *Додати*, а потім знайти та позначити потрібні файли та ще раз скористатися вказівкою *Додати*, тобто підтвердити виконання раніше вибраної вказівки.

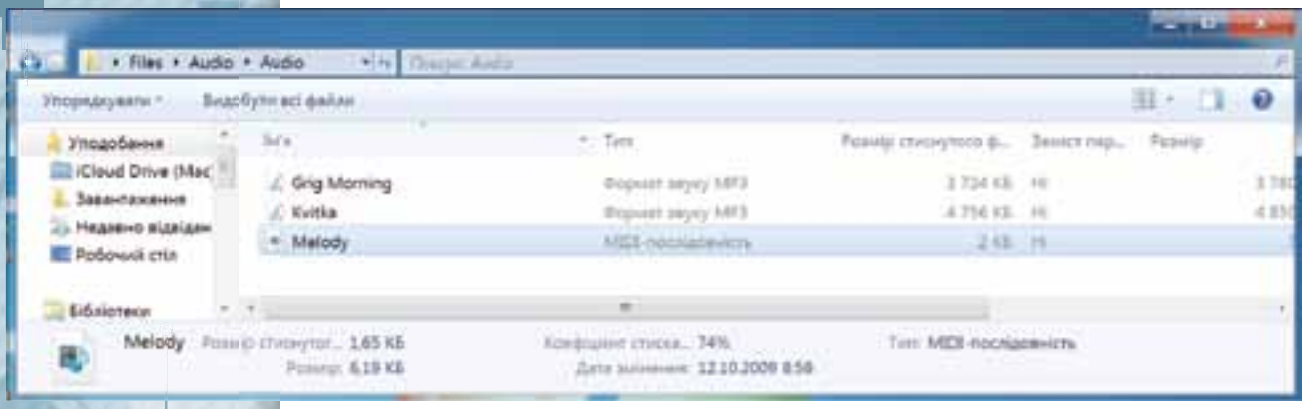
Для видобування з архіву одного або кількох файлів спочатку слід знайти архів. Потім за допомогою програми-архіватора виділити ті файли, які слід розпакувати, та вибрати вказівку розпаковування, вказавши відповідне місце на диску для розміщення файлів, що розкриватимуться. При створенні архіву та занесенні до нього файлів і при його розкриванні залишаються незмінними файли-джерела: при архівуванні — файли, що стискаються; при розкриванні архіву — стиснені файли.

ОС *Windows 7* має вбудовані засоби для роботи із zip-архівами, які ще називають zip-папками. Відрізнити zip-папку від звичайних папок можна за значком — він містить «застібку-блискавку» (мал. 5.14).

Цікаво

Архіви формату ZIP некоректно працюють з іменами файлів і папок, які містять символи кирилиці — при розпакуванні таких архівів імена міститимуть інші символи. Тому при використанні zip-архівів слід іменувати папки та файли латиницею.

Мал. 5.14



Мал. 5.15

Якщо двічі клацнути на такому значку, то можна побачити перелік файлів, які підлягали стисненню, а також у режимі *Таблиці* можна побачити обсяг вихідних файлів та «упакований» обсяг. За необхідності перегляду файлів, що містяться в zip-архіві, можна двічі клацнути на значку відповідного файла. Однак внесення змін у такі документи можливе лише тоді, коли файл розархівовано, або кажуть, файл витягнутий з архіву. Щоб розархівувати файли із zip-архіву, треба його виділити та вибрати вказівку *Видобути всі файли* (мал. 5.15).

У тих випадках, коли архівація виконується для передавання пакета документів іншому користувачеві, слід передбачити наявність у нього програмного засобу, необхідного для розпаковування вихідних даних з архіву. У тому разі, коли користувач не має необхідної програми-архіватора, на основі звичайного архіву можна створити архів, що саморозпаковується, шляхом приєднання невеликого програмного модуля. Файл архіву отримує розширення *exe*, що свідчить про те, що він є виконуваним файлом. Користувач зможе запустити цей файл як звичайну програму, після чого розпакування архіву відбудеться на його комп'ютері автоматично.

Крім того, кожна з програм-архіваторів має багато додаткових функцій.

Цікаво

Архів із розширенням *exe* ще називають SFX-архівом — від англ. *self-extracting archive* — архів, що саморозпаковується.

ДІЄМО

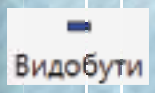
Вправа 3. Архівування папки за допомогою вказівок контекстного меню.
Завдання. Заархівуйте папку *Методи стиснення*, що збережена в папці *Забезпечення комп'ютера*, за допомогою вказівок контекстного меню у форматі *7z*.

1. Відкрийте папку *Забезпечення комп'ютера*, виділіть у ній папку *Методи стиснення* та скопіюйте її до папки *Комп'ютер та програми* власної структури папок.
2. У власній структурі папок клацніть правою кнопкою миші на папці *Методи стиснення* й у контекстному меню оберіть вказівку *7-zip/Додати до "Методи стиснення.7z"*.
3. Переконайтеся, що в папці *Комп'ютер та програми* з'явився відповідний архів.
4. За допомогою вказівки контекстного меню *Властивості* визначте обсяг файла архіву та обсяг вихідної папки, що підлягала архівуванню.

Вправа 4. Розпаковування з архіву окремого файла за допомогою архіватора *7-Zip*.

Завдання. Розпакуйте з архіву *Історія розвитку обчислювальної техніки.7z*, що зберігається в папці *Забезпечення комп'ютера*, файл *Посилання на ресурси в Інтернеті.txt* у власну папку.

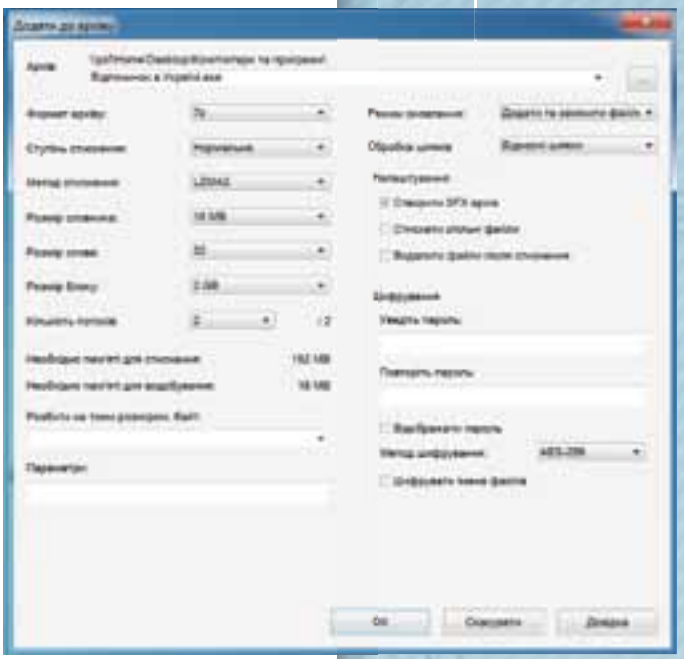
1. Відкрийте папку *Забезпечення комп'ютера*.
2. Двічі клацніть на файлі архіву *Історія розвитку обчислювальної техніки.7z.* Відкриється вікно програми-архіватора *7-Zip* із вмістом зазначеного файла архіву.
3. У вікні програми-архіватора *7-Zip* двічі клацніть на папці *Історія розвитку обчислювальної техніки*, що зберігається в запакованому вигляді.
4. Виділіть файл *Посилання на ресурси в Інтернеті.txt* та натисніть кнопку *Видобути* на панелі інструментів вікна *7-Zip*.
5. У діалоговому вікні *Видобути:*, користуючись інструментом ... для огляду папок, вкажіть папку *Комп'ютер та програми* на своєму комп'ютері, в яку необхідно розпакувати файл, та натисніть кнопку *OK*.
6. Закрийте всі відкриті вікна.



Вправа 5. Створення архіву, що саморозпаковується, за допомогою архіватора 7-Zip.

Завдання. Заархівуйте папку *Відпочинок в Україні* таким чином, щоб отримати архів, що саморозпаковується.

1. Завантажте програму-архіватор *7-Zip*.
2. У списку дисків і папок відкрийте папку *Забезпечення комп'ютера*.
3. Виділіть папку *Відпочинок в Україні* та натисніть кнопку *Додати* на панелі інструментів вікна *7-Zip*.
4. У діалоговому вікні *Додати до архіву* оберіть інструмент ... та вкажіть папку *Комп'ютер та програми*, у яку буде збережено архів. Ім'я архіву за замовчуванням збігатиметься з іменем папки, що додається до архіву.
5. В області *Налаштування* увімкніть прапорець *Створити SFX-архів* і натисніть кнопку *OK* (мал. 5.16).
6. Відкрийте вікно папки, в якій було створено архів, і визначте, які ім'я та розширення має створений файл.
7. Закрийте всі відкриті вікна.



Мал. 5.16

11. У чому полягають основні методи стиснення даних?

Характерною особливістю більшості форматів даних, з якими традиційно працює користувач, є певна надлишковість. Ступінь надлишковості залежить від типу даних.

Прикладом надлишковості є повторення в тексті фрагментів, наприклад, деяких слів або буквосполучень у текстових документах. Подібний надлишок зазвичай усувається заміною повторюваних послідовностей коротшим значенням — кодом. Наприклад, нехай є файл, який містить багато однотипних слів: комп'ютер, комп'ютера, комп'ютерна, комп'ютеризація тощо. Якщо сполучення 9 букв «комп'ютер» замінити

простою комбінацією символів «чц», то розглянутий набір слів перетвориться на систему: «чц», «чца», «чцна», «чцизація» тощо.

Інший вид надлишковості пов'язаний із тим, що деякі значення в даних, що стискаються, трапляються частіше за інші. При цьому можна замінювати дані, які часто трапляються, коротшими кодами, а ті, що трапляються рідко — довшими.

У відеофайлів надлишковість, як правило, в кілька разів менша, ніж у графічних, а в графічних — у кілька разів менша, ніж у текстових. Крім того, ступінь надлишковості даних залежить від прийнятої системи кодування.

Існує велика кількість алгоритмів стиснення даних, але всі вони працюють за одним принципом — зменшення надлишковості даних у файлі за допомогою різних математичних методів. У результаті, залежно від досконалості алгоритму й типу вихідного файла, його розмір може суттєво зменшитися: типовим значенням для документів є 40–50 % і більше. Наднизькі показники у відео- й аудіофайлів. І це цілком логічно, адже дані, що зберігаються в них, піддавалися компресії й практично не містять надлишковості.

Розрізняють такі види стиснення:

- стиснення без втрат, при якому можливе відновлення вихідних даних без спотворень;
- стиснення із втратами — відновлення можливе з незначними спотвореннями.

Стиснення без втрат використовується, зокрема, при опрацюванні та збереженні комп'ютерних програм і даних, коли такі втрати є неприпустимими. Стиснення із втратами зазвичай застосовується для зменшення обсягу звукових, фото-, та відеоданих.

В основі роботи програм-архіваторів лежить процедура пошуку та перекодування однакових фрагментів вмісту файла.

Кожна з програм-архіваторів працює за різними алгоритмами стиснення даних різних типів. У реальних програмах-архіваторах процедура пошуку та перекодування даних відбувається значно складніше.

ЦІКАВІ ДЖЕРЕЛА

Відомості про різні алгоритми стиснення та програми для архівування даних:
http://wiki.tntu.edu.ua/Архівація

ОБГОВОРЮЄМО

Обговоріть запитання, що містяться у файлі *Тема 5*, який зберігається у папці *Обговорюємо*. Також відкрити ці запитання можна за допомогою QR-коду.

ПРАЦЮЄМО В ПАРАХ

1. Із переліку програм (мал. 5.17) складіть «піраміду», що відображатиме порядок встановлення зазначених програм на комп'ютері для пошуку необхідних даних в Інтернеті та створення на їх основі текстового повідомлення на задану тему. Обговоріть у парах можливі варіанти виконання цього завдання.

Драйвер модема

Прикладні програми забезпечення доступу до Інтернету

Антивірусні програми

Операційна система

Текстовий редактор

Мал. 5.17

2. Як розкрити раніше створений архів? Сформулюйте узагальнене правило розпаковування архіву. Обговоріть його в парах.

3. Чому в певні комп'ютерні ігри не можна пограти на деяких комп'ютерах? Сформулюйте три можливі причини. Обговоріть у парах.

ПРАЦЮЄМО САМОСТІЙНО

1. Доповніть класифікаційну схему програмного забезпечення комп'ютера, збережену в папці *Забезпечення комп'ютера* у файлі *Програмне забезпечення,* прикладами програм із різними ліцензіями. За потреби скористайтеся відомостями з Інтернету.
2. Знайдіть в Інтернеті відомості про популярність сучасних операційних систем. На основі числових даних у табличному процесорі побудуйте діаграми, які ілюструватимуть знайдені відомості. Зробіть висновок щодо популярності операційної системи, яку ви використовуєте для персонального комп'ютера та мобільного пристрою.
3. У списку інстальованих на вашому комп'ютері програм визначте три програми, які ви використовуєте дуже часто, і дві, які використовуєте зрідка. Встановіть, який обсяг носія даних буде звільнено, якщо деінсталювати одну із програм, яку ви зовсім не використовуєте. Знайдіть в Інтернеті відомості про призначення цієї програми та зробіть висновок про доцільність чи недоцільність її деінсталяції.
4. При архівуванні даних використовують різні методи стиснення. Одним із них є алгоритм Хаффмана. В основі алгоритму Хаффмана лежить ідея кодування бітовими групами. Розглянемо простий приклад, що ілюструє роботу алгоритму Хаффмана. Нехай задано текст, у якому літера *A* трапляється 10 разів, літера *B* — 8 разів, *C* — 6 разів, *D* — 5 разів, *E* і *F* — по 4 рази. Тоді один із можливих варіантів кодування за алгоритмом Хаффмана наведено в таблиці 5.1.

Таблиця 5.1

Символ	Частота входження	Бітовий код	Символ	Частота входження	Бітовий код
A	10	00	*D*	5	101
B	8	01	*E*	4	110
C	6	100	*F*	4	111

Складіть таблиці для кодування за алгоритмом Хаффмана слів «молоко», «кукуріку». Знайдіть в Інтернеті додаткові відомості про алгоритм Хаффмана та з'ясуйте, як визначити довжину коду заданого слова та коефіцієнт стиснення даних.

ДОСЛІДЖУЄМО

Дослідіть, які дії необхідно виконати для деінсталяції програми. Для цього встановіть та деінсталюйте пробну версію програми *Easy GIF Animator* за таким планом:

1. Завантажте пробну версію програми *Easy GIF Animator* із сайта *http://easy-gif-animator.ru.uptodown.com* та інсталюйте її на комп'ютер.
2. Відкрийте *Головне меню* й оберіть вказівку *Панель керування*.

3. У списку категорій груп програм оберіть *Програми/Програми та засоби.*
4. Зачекайте, поки не буде сформовано список інстальованих програм на комп'ютері. Відшукайте у списку програму *Easy GIF Animator v 6.2*, клацніть на її назві у списку.
5. У списку вказівок, доступних для цієї програми, оберіть вказівку *Видалити*. Підтвердіть бажання видалити програму з комп'ютера, натиснувши кнопку з написом *Так*.
6. Дочекайтеся завершення процесу деінсталяції. Переконайтеся, що назва деінстальованої програми не відображається ані у списку програм, ані в головному меню комп'ютера.

6. ПРАКТИЧНА РОБОТА 3

АРХІВУВАННЯ ТА РОЗАРХІВУВАННЯ ДАНИХ

ПРИГАДАЙТЕ

- як створювати архіви різних типів;
- як додавати дані до архівів, знаходити дані в архівах, вилучати дані з архівів;
- як оновлювати архіви;
- як архівувати та розархівовувати файли та папки.

СТВОРІТЬ

У власній структурі папок створіть папку *Практична робота 3.*

ПАМ'ЯТАЙТЕ

Під час виконання практичних завдань пам'ятайте про правила безпеки життєдіяльності при роботі з комп'ютером!

Завдання 1. Відкриття файла з архіву (*5 балів*)

Відкрийте файл архіву з демонстраційною програмою Міжнародного конкурсу з інформатики та комп'ютерної вправності «Бобер» (*http://bober.net.ua/*) 2014 року. Зробіть висновок, чи можна це зробити з вікна програми *7-Zip*, не розпаковуючи архіву.

Завдання 2. Видалення файла із архіву (*4 бали*)

В архіві *Пісні Марусі Чурай*, який збережений у папці *Забезпечення комп'ютера*, видаліть файл із текстом, що не належить авторці.

Завдання 3. Додавання файлів до архіву (*4 бали*)

Використовуючи інструменти програми *7-Zip*, додайте до архіву *Історичні пісні*, який збережений у папці *Забезпечення комп'ютера*, файл *Пісня про Байду.*

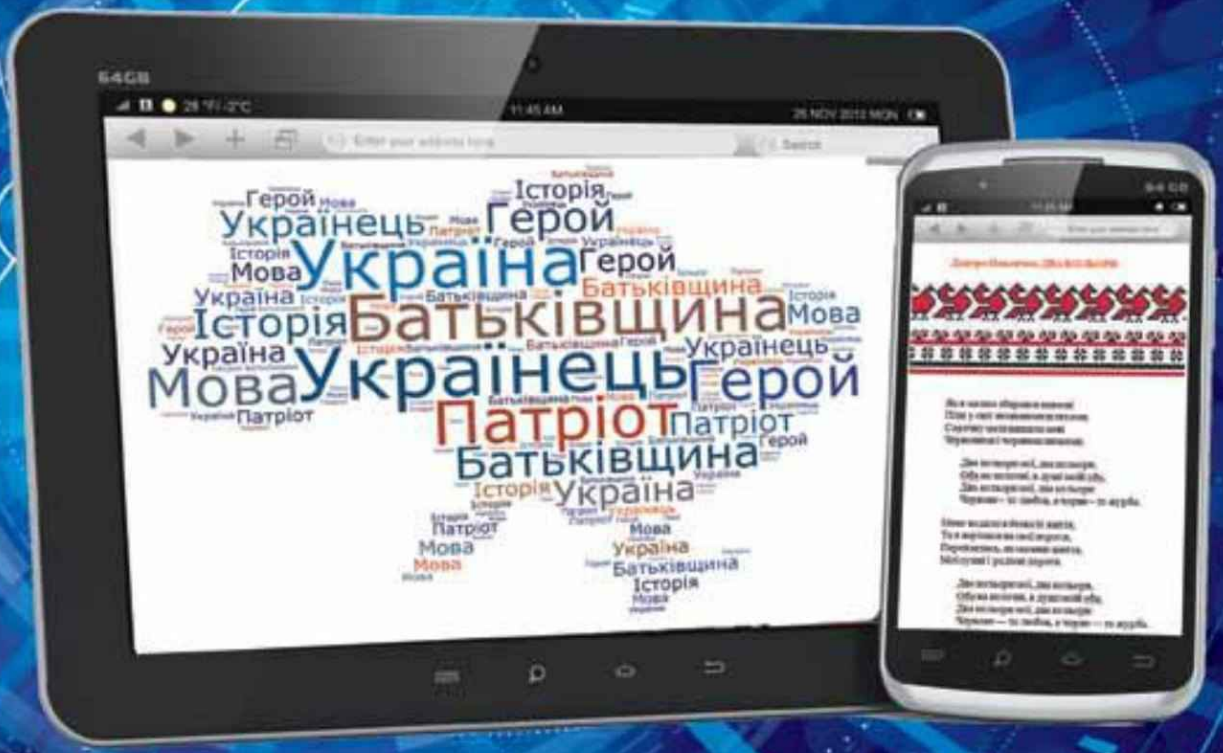
Завдання 4. Створення архівів різних типів (*5 балів*)

Для файла *Запорозьке козацтво*, який збережений у папці *Забезпечення комп'ютера*, створіть rar-, zip- та SFX-архіви. Порівняйте розміри утворених архівів.

Завдання 5. Оновлення файла в архіві (*6 балів*)

Доповніть презентацію в архіві *Інструменти програми-архіватора*, збереженому в папці *Забезпечення комп'ютера*, відомостями про призначення інструментів програми *7-Zip.*

РОЗДІЛ 3



ОПРАЦЮВАННЯ ТЕКСТОВИХ ДАНИХ

7. ТЕКСТОВИЙ ДОКУМЕНТ І ЙОГО ОБ'ЄКТИ

ПРИГАДАЙТЕ:

- як створювати, відкривати та зберігати текстовий документ;
- як вводити текст із клавіатури з дотриманням правил орфографії, пунктуації;
- як виділяти та вилучати, копіювати й переміщувати фрагменти тексту;
- як форматувати символи та абзаци тексту;
- як вставляти графічні об'єкти в текстовий документ.

ВИ ДІЗНАЄТЕСЯ:

- які існують формати файлів текстових документів;
- як до тексту вставити символи, яких немає на клавіатурі;
- які типи списків можна створити в текстовому документі;
- як розмістити текст у кілька колонок;
- як додати формули до текстового документа;
- як створювати та форматувати графічні об'єкти в середовищі текстового процесора;
- як до текстового документа вставити таблиці;
- як редагувати та форматувати структуру таблиці;
- що таке недруковані знаки та як їх переглянути.

ВИВЧАЄМО

1. Які існують формати файлів текстових документів?

Ви вже знаєте, що імена текстових файлів, що створені в різних текстових редакторах чи процесорах, можуть мати різні розширення. Наприклад, документ, створений у текстовому редакторі *Блокнот*, має розширення *txt*, у текстовому процесорі *Microsoft Word* — *doc* або *docx*, у *LibreOffice Writer* — *odt*. Розширення файлів відповідають **форматам**, які визначають спосіб організації даних у файлі. Файли різних форматів можуть мати різні значки. Це залежить від того, які програми встановлені на комп'ютері за замовчуванням для роботи з файлами відповідних форматів (мал. 7.1).

Мал. 7.1

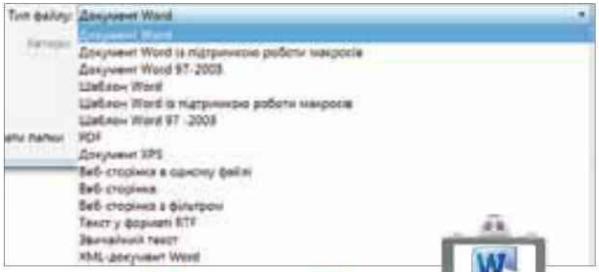
Текстовий файл може містити як форматований, так і неформатований текст.

Стандартним форматом для збереження неформатованого тексту є *Звичайний текст*, файли цього формату мають розширення *txt*. Усі дані в такому файлі представлено символами кодової таблиці, які без будь-яких перетворень можна вводити з клавіатури, виводити на екран чи принтер.

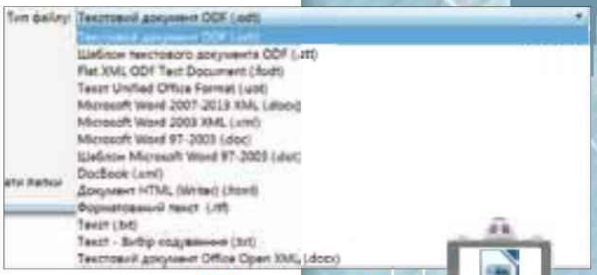
Текстові процесори та видавничі системи використовують спеціально розроблені формати файлів, що містять не тільки текст, а й відомості про те, як він має бути оформлений: які параметри форматування слід застосувати до певних символів, фрагментів тексту чи абзаців та які інші об'єкти мають бути розташовані разом із текстом. Ці додаткові відомості носять назву **розмітки тексту**.

Універсальним форматом для збереження форматованого тексту, що підтримується різними текстовими процесорами, є формат *RTF* (від англ. *Rich Text Format* — розширений текстовий формат).

Текстові процесори підтримують роботу з документами кількох форматів. Під час збереження текстового документа в середовищі текстового



Мал. 7.2, *а*



Мал. 7.2, *б*

процесора крім імені файла вказується також його тип, що визначає формат файла текстового документа. У списку *Тип файлу*, що розкривається, вказано перелік доступних форматів. Першим, як правило, вказано формат, який є власним форматом обраного текстового процесора: *Документ Word* у *Microsoft Word* (мал. 7.2, *а*) та *Текстовий документ ODF* у *LibreOffice Writer* (мал. 7.2, *б*).

Для перегляду готових текстових документів використовують формати *PDF* та *Веб-сторінки* — з розширеннями *htm, html* (від англ. *HyperText Markup Language* — мова розмітки гіпертекстових документів).

Текстові документи у форматі веб-сторінки можна переглядати за допомогою браузера. Їх відмінністю від звичайного текстового файла є те, що в HTML-документах використовують спеціальні команди — теги, які визначають правила форматування документа. Проте текстові документи, збережені у форматі веб-сторінки, не завжди будуть відображені саме в такому вигляді, як вони були створені — це залежить від розмірів монітора, налаштувань браузера та інших властивостей. У разі потреби переглянути публікацію саме такою, як вона була створена, використовують формат PDF.

Цікаво

Portable Document Format (PDF) — відкритий формат файла, створений і підтримуваний компанією *Adobe Systems* для представлення документів у незалежному від пристрою виведення та роздільної здатності вигляді. У грудні 2007 р. формат PDF було затверджено як міжнародний стандарт.

2. Як до тексту вставити символи, яких немає на клавіатурі?

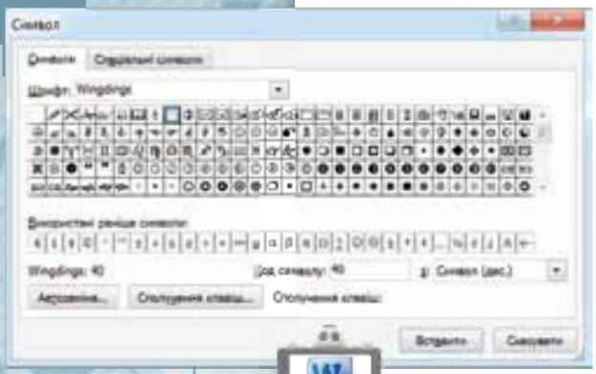
Ви вже вмієте створювати текстові документи, що містять різні об'єкти: символи, що вводяться з клавіатури, абзаци, графічні зображення, а також редагувати та форматувати їх.

Іноді до тексту необхідно додати символи, яких немає на клавіатурі. Це можуть бути знаки авторського права ©, математичних операцій порівняння ≠, ≤, ≥, ≈, грецькі літери α, β та інші символи.

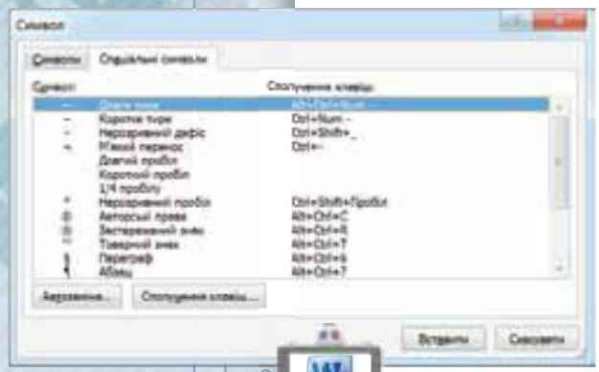
Щоб додати до документа такі символи, у *Microsoft Word 2010* використовують інструмент *Символ* на вкладці *Вставлення* групи *Символи* (мал. 7.3). Деякі часто вживані символи відображені у списку, що розкривається. За допомогою вказівки *Інші символи* відкривається вікно *Символ* (мал. 7.4), у якому можна обирати різні символи. Список *Шрифт* містить перелік шрифтів, які можна обирати. Наприклад, шрифт *Wingdings* містить символи у вигляді малюнків книги, телефона, листа, папки тощо (мал. 7.4).

Мал. 7.3

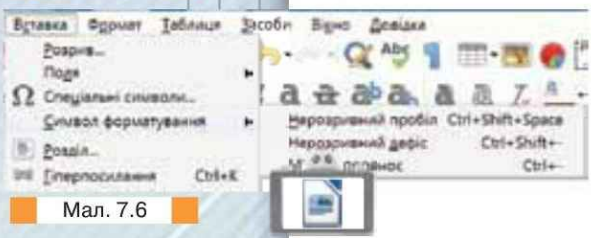
Деякі символи можна вводити за допомогою комбінацій клавіш або обрати на вкладці *Спеціальні символи* вікна *Символ* (мал. 7.5). Зокрема, символи *нерозривний дефіс* (*Ctrl+Shift+ –*) і *нерозривний пропуск* (*Ctrl+Shift+Пропуск*) використовують тоді, коли слово або фразу, що містить дефіс або пропуск, слід сприймати як одне слово й не розривати на



Мал. 7.4



Мал. 7.5



Мал. 7.6

два рядки. Наприклад, щоб ініціали та прізвище *І. В. Карпенко* завжди були записані в одному рядку, слід після ініціалів використати не звичайний пропуск, а нерозривний пропуск.

У текстовому процесорі *LibreOffice Writer* вставити символи можна за допомогою інструмента *Спеціальні символи* Ω на панелі інструментів *Стандартна* або вказівки меню *Вставка/Спеціальні символи*. Вікно *Вибір символу* схоже на вікно *Символ* у *Microsoft Word*. Нерозривний дефіс чи нерозривний пропуск можна вставити за допомогою комбінацій клавіш або вказівки *Вставка/Символ форматування* (мал. 7.6).

Для вставлених символів, як і для символів, що вводяться з клавіатури, можна змінювати значення параметрів форматування — колір, розмір, накреслення тощо.

3. Які типи списків можна створити в текстовому документі?

Деякі набори даних, які містяться в документі, можна оформити у вигляді списку, наприклад, кроки інструкції виконання завдання, перелік обов'язків чергового учня, складений план художнього твору. Текстові процесори містять інструменти, за допомогою яких можна створювати **нумеровані**, **марковані** та **багаторівневі** списки. Нумерований список створюється, якщо потрібно пронумерувати кожний елемент списку — це відбувається автоматично, маркований — якщо нумерація не важлива, а слід лише показати, що це окремий елемент списку.

Якщо елементи списку вже введені в документі й кожний із них розміщується в окремому абзаці, то для створення списку потрібно лише виділити їх та обрати відповідний інструмент. У текстовому процесорі *Microsoft Word 2010* інструменти для створення списків розташовані на вкладці *Основне* в групі *Абзац*, у текстовому процесорі *LibreOffice Writer* — на панелі інструментів *Форматування*. У різних текстових процесорах інструменти для створення списків мають схожий вигляд та призначення: *Маркери* (*Маркований список*), *Нумерація* (1 2 3) чи *Багаторівневий список*. У списку, що відкриється при виборі одного з інструментів, відображаються елементи відповідної бібліотеки (мал. 7.7–7.9). Щоб обрати потрібний елемент, достатньо клацнути на ньому лівою кнопкою миші.

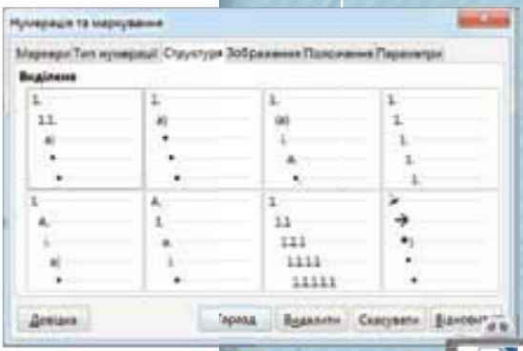
Мал. 7.7

Мал. 7.8

Мал. 7.9

Щоб змінити параметри форматування списків у текстовому процесорі *Microsoft Word 2010*, потрібно обрати вказівку *Визначити новий маркер* або *Визначити новий числовий формат* у нижній частині вікна відповідної бібліотеки.

Змінити параметри форматування списків або створити багаторівневий список у текстовому процесорі *LibreOffice Writer* можна за допомогою інструментів, розташованих на вкладках вікна *Нумерація і маркування* (мал. 7.10). Відкрити його можна за допомогою вказівки *Ще маркери* або *Ще нумерація* в нижній частині вікна бібліотеки маркерів чи нумерації. Обрати формати багаторівневого списку можна на вкладці *Структура* (мал. 7.10).



Мал. 7.10

Створити новий список, елементи якого ще не введені до текстового документа, можна аналогічно: спочатку необхідно встановити курсор на новий рядок, обрати відповідний інструмент для створення списку, ввести дані та натиснути клавішу *Enter*. У новому абзаці створюється новий елемент списку. Для завершення списку в новому абзаці, який не має бути елементом списку, слід ще раз обрати інструмент, за допомогою якого створювався список, щоб відмовитися від нумерації чи маркування.

ДІЄМО

Вправа 1. Створення нумерованого та маркованого списків.
Завдання. У документі *Підготовка реферату,* що зберігається в папці *Текстовий процесор,* створіть нумерований список для списку друкованих видань та маркований список для списку основних понять, що розглядаються в рефераті.

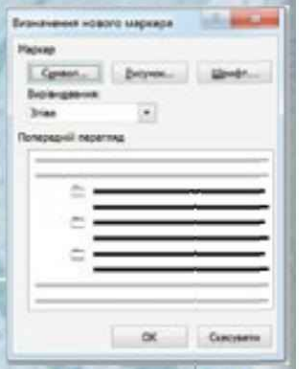
1. У власній структурі папок створіть папку *Тексти*.
2. Відкрийте документ *Підготовка реферату*, що зберігається в папці *Текстовий процесор*.

3. Виділіть елементи списку друкованих видань й оберіть інструмент *Нумерація* та формат нумерації з крапкою після номера (мал. 7.8).

4. Для списку друкованих видань цього документа внесіть такі зміни.
   - Видаліть елементи списку, які дублюються. Проаналізуйте, як змінилися номери інших елементів такого нумерованого списку.
   - Упорядкуйте список джерел в алфавітному порядку. Для переміщення елементів списку застосуйте різні способи: перетягування мишею, використання буфера обміну. Проаналізуйте, як змінюються номери елементів списку.

5. Для списку основних понять цього документа змініть вигляд маркера на символ папки, для цього в бібліотеці маркерів оберіть вказівку *Визначити новий маркер* у *Microsoft Word 2010* (вказівку *Ще маркери* в *LibreOffice Writer*), натисніть кнопку *Символ* (мал. 7.11), оберіть шрифт *Wingdings,* символ папки та натисніть кнопку *OK*. Установіть такі значення для форматування елементів списку: відступ розташування маркера — 0,5 см; відступ розташування тексту — 1,1 см.

6. Збережіть отриманий документ у файлі з тим самим іменем у папці *Тексти* власної структури папок.



Мал. 7.11

4. Як розмістити текст у кілька колонок?

У газетах і журналах вам траплявся текст, розбитий на кілька колонок. Такий текст можна підготувати в середовищі текстового процесора, при цьому різні абзаци можна розбити на різну кількість колонок.

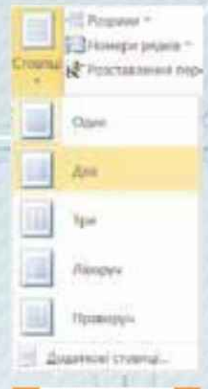
Процес підготовки документа, що містить колонки, складається з двох етапів: спочатку вводять текст, а потім виділяють потрібні абзаци та вказують кількість колонок, у які слід розмістити виділений фрагмент.

У *Microsoft Word 2010* для цього використовують інструмент *Стовпці* на вкладці *Розмітка сторінки* в групі *Параметри сторінки*. Користувачеві пропонується п'ять варіантів розташування виділеного тексту в шпальти (мал. 7.12).

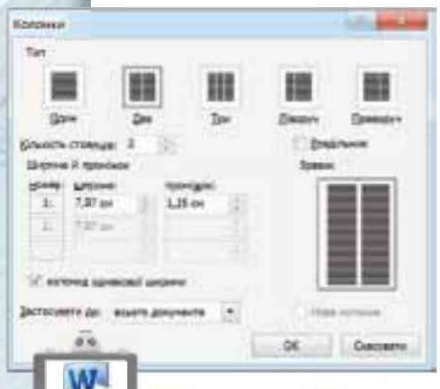
Для форматування колонок використовують вказівку *Додаткові стовпці,* при цьому відкривається вікно *Колонки* (мал. 7.13). Усі власти-

Цікаво

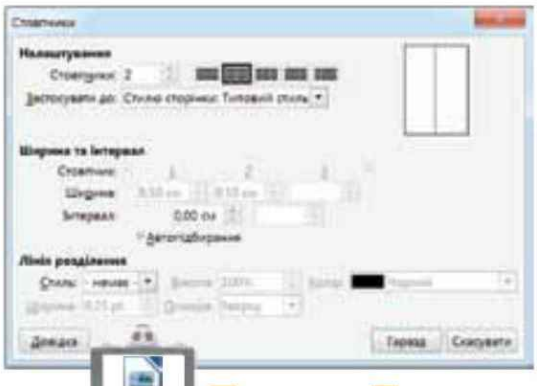
Шпальта — стовпчик друкованого тексту в газеті, журналі тощо.



Мал. 7.12



Мал. 7.13



Мал. 7.14

вості налаштування інтуїтивно зрозумілі, до того ж в області *Зразок* відображається схематичний вигляд сторінки.

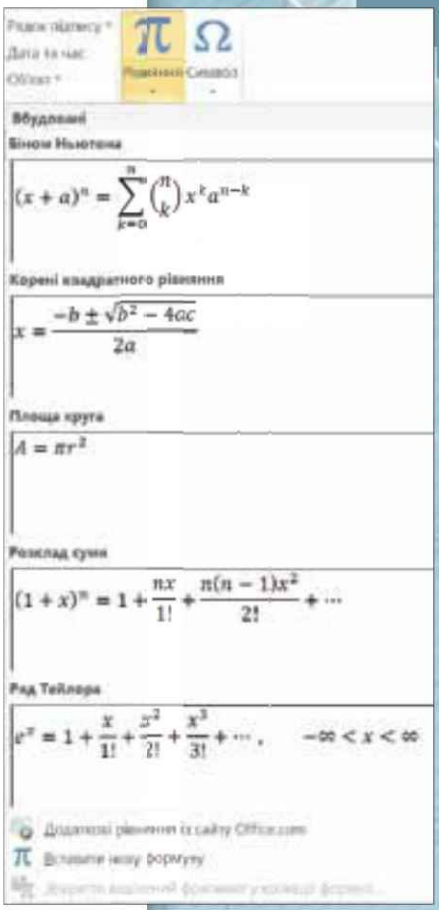
У *LibreOffice Writer* для створення колонок використовують вказівку *Формат/Колонки*. При цьому з'являється вікно *Стовпчики* (мал. 7.14), у якому можна обрати кількість колонок та параметри їх форматування.

ДІЄМО

Вправа 2. Стінна газета.

Завдання. Подайте текст *Пам'яті героїв Крут,* що зберігається у файлі *Історія* папки *Текстовий процесор*, для розміщення на інформаційному історичному стенді у школі за таким макетом:

1. Завантажте текстовий документ *Історія,* що зберігається в папці *Текстовий процесор.*
2. Виділіть текст на першій сторінці. Скористайтесь інструментом *Стовпці* на вкладці *Розмітка сторінки* в групі *Параметри сторінки* текстового процесора *Microsoft Word 2010* або вказівкою *Формат/Колонки* в *LibreOffice Writer*. Оберіть спосіб формування шпальт — *Ліворуч*. Завершіть дію натискуванням кнопки *ОК* (*Гаразд*).
3. Виконайте попередню дію для другої та третьої сторінок тексту, щоразу обираючи інший спосіб формування шпальт, як показано в макеті.
4. Збережіть файл з іменем *Стенд* у папці *Тексти* у стандартному форматі текстового процесора та у форматах *PDF* і *RTF*. Перевірте, чи впливає обраний формат на спосіб розміщення тексту в колонки та як зміна формату впливає на розмір файла. Зробіть висновки.



Мал. 7.15

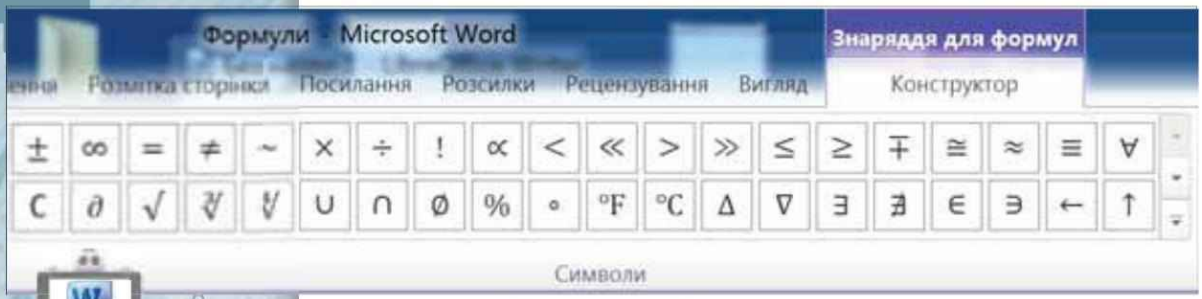
5. Як додати формули до текстового документа?

Текстові процесори містять вбудовані засоби для створення математичних, фізичних чи хімічних формул, які не можна ввести з клавіатури. Це спрощує процес створення навчальних і наукових документів, що містять формули.

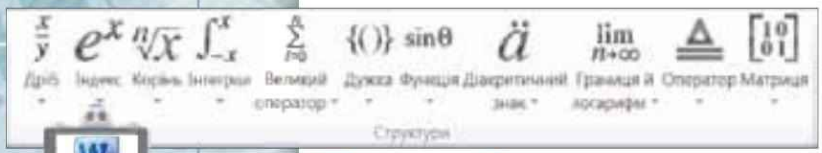
У *Microsoft Word 2010* для цього призначений інструмент *Рівняння* на вкладці *Вставлення* в групі *Символи*. У списку, що розкривається, наведені деякі вбудовані формули, а також доступна вказівка *Вставити нову формулу* (мал. 7.15), за допомогою якої можна створювати власні формули.

Після вибору цієї вказівки стають доступними інструменти *Конструктора* для роботи з формулами, за допомогою яких можна вставляти до поля формули спеціальні символи (мал. 7.16) та структури: дроби, індекси, корені, дужки тощо (мал. 7.17).

Список, що розкривається при виборі інструмента для додавання певної структури, містить набір різних шаблонів — зразків фрагмента формули з використанням обраної структури. Наприклад, за допомогою

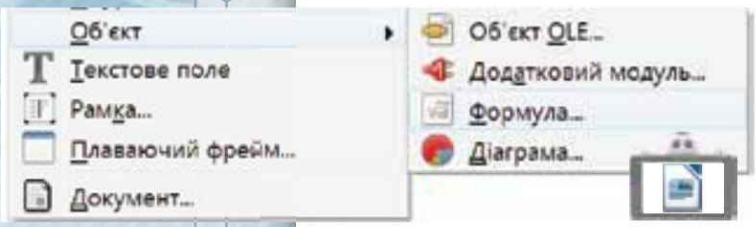


Мал. 7.16

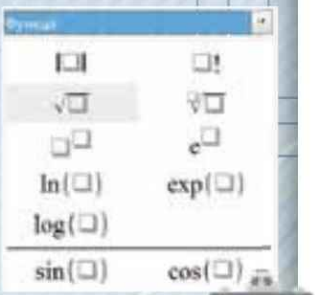


Мал. 7.17

Мал. 7.18



Мал. 7.19



Мал. 7.20

інструмента *Корінь* можна обрати шаблони для запису квадратного чи кубічного кореня, а також кореня, степінь якого необхідно ввести з клавіатури (мал. 7.18).

Для конструювання формули слід обрати необхідні структури й у виділені поля ввести потрібні символи. Символи можна вводити з клавіатури або обирати серед символів на вкладці *Конструктор*. Для завершення створення формули слід клацнути лівою кнопкою миші за її межами — тобто завершити використання спеціального засобу.

Щоб відредагувати створену формулу, необхідно двічі клацнути мишею в її межах, при цьому стануть доступними інструменти створення та редагування формули на вкладці *Конструктор*.

Для додавання формули в текстовий документ у *LibreOffice Writer* використовують вказівку *Вставка/Об'єкт/Формула* (мал. 7.19).

Символи та структури, які можна використовувати у формулах, зібрані в список, що відкривається в лівій частині вікна (мал. 7.20), та згруповані за категоріями: відношення, функції, фігурні дужки тощо.

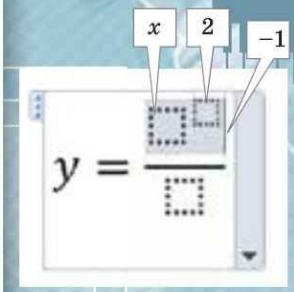
ДІЄМО

Вправа 3. Створення формули.
Завдання. Створіть текстовий документ *Формула*, до якого вставте формулу

$$y = \frac{x^2 - 1}{|x| - 1}.$$

1. Завантажте текстовий процесор та створіть новий текстовий документ.
2. На вкладці *Вставлення* оберіть інструмент *Рівняння* в групі *Символи*.

3. Введіть із клавіатури початок формули: *y*=.
4. На вкладці *Конструктор* у групі *Структури* оберіть інструмент *Дріб* і шаблон дробу з горизонтальною рискою .
5. Виділіть чисельник дробу, на вкладці *Конструктор* у групі *Структури* оберіть інструмент *Індекс* і шаблон у вигляді степеня.
6. Виділіть основу степеня, введіть символ *x*, виділіть показник степеня, введіть символ *2*, встановіть вказівник миші в чисельнику дробу та введіть *–1* (мал. 7.21).
7. Виділіть знаменник дробу. На вкладці *Конструктор* у групі *Структури* оберіть *Дужка* та шаблон у вигляді модуля . Виділіть об'єкт під модулем, введіть символ *x*, встановіть вказівник миші у знаменнику після модуля, введіть *–1*.
8. Клацніть мишею за межами формули. Перевірте, що створена формула відповідає зразку.
9. Збережіть результат у файлі з іменем *Формула* в папці *Тексти* власної структури папок.



Мал. 7.21

6. Як створювати та форматувати графічні об'єкти в середовищі текстового процесора?

Ви вже вмієте додавати до текстового документа готові графічні зображення та форматувати їх. Текстові процесори містять вбудовані засоби, за допомогою яких можна створювати зображення, що складаються з ліній, прямокутників та інших фігур, безпосередньо в середовищі текстового процесора.

У *Microsoft Word 2010* для створення зображень використовують інструмент *Фігури* з групи *Зображення* на вкладці *Вставлення*, у *LibreOffice Writer* — вказівку *Фігури* в меню *Вставка*.

Microsoft Word 2010 містить велику бібліотеку готових фігур, які використовують для створення зображень, а також відповідні інструменти в середовищі графічного редактора (мал. 7.22, *а*). Для створення зображень у *LibreOffice Writer* можна скористатися фігурами *Лінія*, *Прямокутник*, *Еліпси* та *Багатокутник* (мал. 7.22, *б*).

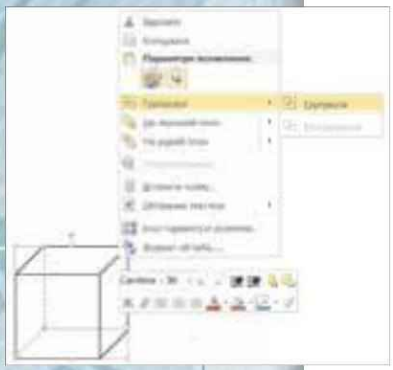
Мал. 7.22, *а*

Мал. 7.22, *б*

Вставлені зображення можна форматувати: змінювати властивості заливки фігури та її контуру, обирати ефекти та стилі фігур тощо. Значення цих властивостей у *Microsoft Word 2010* можна задавати за допомогою інструментів на вкладці *Формат* в області *Засоби креслення*, яка з'являється, якщо виділити створену фігуру.

При розміщенні зображення в текстовому документі разом із текстом, як і для готових малюнків, важливим є спосіб розташування зображення відносно тексту. Обрати його можна за допомогою інструментів *Розташування* та *Обтікання текстом* із групи *Упорядкування* вкладки *Формат*. Інструменти з груп *Упорядкування* та *Розмір* використовують також для повороту, вирівнювання, вказування порядку розміщення кількох фігур та налаштування розміру фігури. Зображення, які складаються з кількох фігур, можна згрупувати в один об'єкт. Для цього спо-

Мал. 7.23



Мал. 7.24

чатку слід при натисненій клавіші *Shift* послідовно виділити всі фігури, що входять до складу зображення. Потім необхідно виконати вказівку *Групувати* за допомогою відповідного інструмента з групи *Упорядкування* вкладки *Формат* або контекстного меню (мал. 7.24). Коли кілька фігур згруповані в один об'єкт, тоді для складеного зображення легко змінювати розміри та місце розміщення в текстовому документі, а також інші властивості.

У *LibreOffice Writer* зображення можна форматувати за допомогою інструментів на панелі *Властивості рисунка* (мал. 7.23), яка з'являється замість панелі *Форматування*, якщо виділити створену фігуру.

Контекстне меню виділеної фігури чи групи фігур у *LibreOffice Writer* містить вказівки для зміни розташування об'єкта, вирівнювання, обтікання текстом, групування кількох виділених об'єктів тощо.

ДІЄМО

Вправа 4. Створення зображення куба.

Завдання. У новому документі *Куб* засобами текстового процесора створіть зображення куба. Додайте текстові написи для позначення вершин куба.

1. У папці *Тексти* власної структури папок створіть новий документ з іменем *Куб*.
2. На вкладці *Вставлення* оберіть інструмент *Фігури* й зі списку фігур у категорії *Основні фігури* оберіть *Куб*. Виконайте протягування мишею в тому місці документа, де ви плануєте вставити зображення.
3. Виділіть вставлене зображення й на вкладці *Формат* у групі *Стилі фігур* оберіть інструмент *Заливка фігури*. Встановіть значення цієї властивості *Без заливки* (мал. 7.25), щоб усі грані куба не містили заливки.
4. Серед фігур на вкладці *Вставлення* оберіть *Лінія* та додайте до зображення куба три лінії, які відображають невидимі ребра куба.
5. Послідовно виділіть кожну з ліній і за допомогою інструмента *Контур фігури* із вкладки *Формат* змініть значення властивості *Штрихи* на пунктирну лінію.
6. Натисніть на клавіатурі клавішу *Shift*, утримуйте її й виділіть лінії та фігуру, з яких складається куб. Клацніть правою кнопкою миші на одній з виділених меж і в контекстному меню оберіть вказівку *Групувати/Групувати* (мал. 7.24).
7. За допомогою інструмента *Напис* із вкладки *Вставлення* додайте поруч із вершинами куба написи з іменами вершин: *A*, *B*, *C*, *D*, *A'*, *B'*, *C'*, *D'*.
8. Збережіть результати роботи.

Мал. 7.25

7. Як до текстового документа вставити таблиці?

Для подання даних у структурованому вигляді в текстових документах використовують таблиці. Ті відношення між даними, які можуть бути не помічені в разі запису «в один рядок», стають наочними при поданні у вигляді

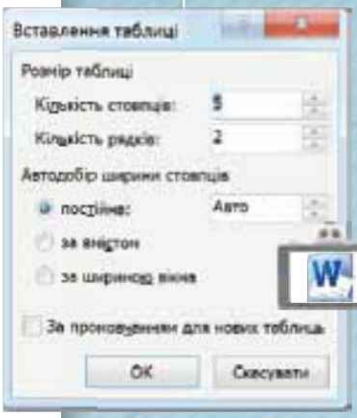
таблиці. Наприклад, у вигляді таблиці зручно подавати дані про проведення дослідів за різних умов на уроках фізики.

Як і при роботі з електронними таблицями, у середовищі текстового процесора при роботі з таблицями розрізняють такі об'єкти: таблиця, рядки, стовпці, клітинки. Кожний із об'єктів має властивості та список дій, які можна з ним виконувати.

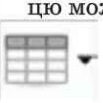
Для вставлення таблиці до текстового документа в *Microsoft Word 2010* використовують інструмент *Таблиця* на вкладці *Вставлення*. З його допомогою можна створювати таблицю різними способами:

1. Виділити мишею кількість рядків і стовпців таблиці (мал. 7.26).
2. Обрати вказівку *Вставити таблицю* та вказати потрібну кількість стовпців і рядків у вікні *Вставлення таблиці* (мал. 7.27).
3. За допомогою вказівки *Намалювати таблицю* можна легко створити таблицю складної структури, наприклад таку, яка містить клітинки різної висоти або різну кількість стовпців на рядок. При цьому використовується метод, схожий із малюванням таблиці від руки. Вказівник миші перетворюється на олівець. За допомогою цього олівця можна намалювати межі клітинок, рядків і стовпців — так само, як на аркуші паперу.

Мал. 7.26



Мал. 7.27

Аналогічно в текстовому процесорі *LibreOffice Writer* таблицю можна додати за допомогою інструмента *Вставити таблицю*  (мал. 7.28) або вказівки *Таблиця/Вставити таблицю*, що викликає вікно *Вставити таблицю*.

Мал. 7.28

8. Як редагувати та форматувати структуру таблиці?

Після створення таблиці її можна форматувати. Можна також вносити зміни до структури таблиці: об'єднувати клітинки, вставляти й видаляти стовпці чи рядки або змінювати їх розмір.

Якщо необхідно застосувати однакові параметри форматування до всіх клітинок таблиці, її слід спочатку виділити. Щоб швидко виділити всю таблицю, слід навести вказівник миші на будь-яку клітинку таблиці та клацнути на позначці , яка з'являється поруч із лівим верхнім кутом таблиці.

Коли текстовий курсор розташовано в одній із клітинок таблиці, стають доступними інструменти для редагування й форматування таблиці та її об'єктів.

У *Microsoft Word 2010* для цього можна використовувати різні інструменти в області *Табличні знаряддя* на вкладках *Конструктор* і *Макет* або вказівки контекстного меню.

За допомогою інструментів із вкладки *Конструктор* (мал. 7.29) можна обрати стиль таблиці, змінити значення параметрів форматування таблиці — колір і стиль заливки, встановити обрамлення окремих виділених клітинок, змінити товщину та колір межі тощо. Налаштування цих властивостей схоже на відповідні дії в електронних таблицях.

Мал. 7.29

Редагувати структуру таблиці — вставляти й видаляти рядки чи стовпці, об'єднувати або розділяти клітинки — можна за допомогою інструментів вкладки *Макет* (мал. 7.30) та вказівок контекстного меню. Для об'єднання клітинок таблиці їх потрібно виділити, потім обрати відповідний інструмент на вкладці *Макет* або вказівку *Об'єднати клітинки* в контекстному меню. За допомогою інструмента *Напрямок тексту* можна розміщувати текст у клітинках горизонтально або вертикально.

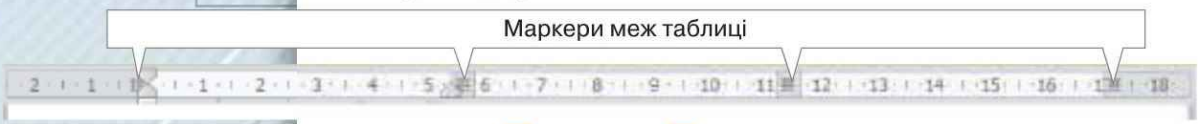
Мал. 7.30

У *LibreOffice Writer* редагування й форматування структури таблиці та її об'єктів можна виконати за допомогою інструментів на панелі *Таблиці* (мал. 7.31), вказівок меню *Таблиця* або контекстного меню.

Мал. 7.31

Спочатку всі стовпці мають однакову ширину, а рядки — однакову висоту. Можна вручну змінити ширину окремого стовпця або дозволити системі автоматично вибрати ширину кожного стовпця залежно від його вмісту.

Змінити ширину стовпців і висоту рядків можна, переміщуючи маркери меж таблиці на горизонтальній чи вертикальній лінійці масштабування (мал. 7.32).



Мал. 7.32

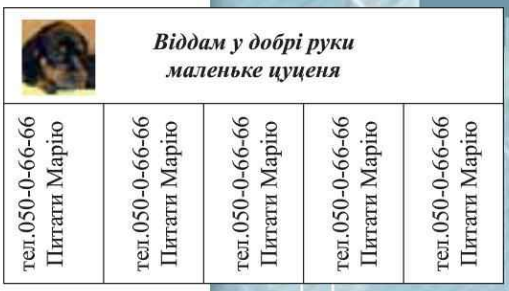
Для зміни розмірів стовпців чи рядків можна також навести вказівник миші на відповідну межу в таблиці та, коли він набуде вигляду лінії з двонапрямленою стрілкою, виконати протягування мишею в потрібному напрямку.

ДІЄМО

Вправа 5. Створення таблиці та налаштування її властивостей.
Завдання. Створіть документ з оголошенням про пошук господарів для цуценяти з використанням таблиці та зображення (мал. 7.33).

1. У папці *Тексти* власної структури папок створіть новий документ, вмістом якого буде оголошення про пошук господарів для цуценяти.

2. У створеному документі вставте таблицю 2 × 5 одним із відомих вам способів.
3. Об'єднайте всі клітинки першого рядка таблиці.
4. Введіть у клітинці першого рядка текст «Віддам у добрі руки маленьке цуценя» та застосуйте такі параметри форматування: шрифт — *Times New Roman*, розмір — *12*, стиль накреслення — *напівжирний, курсив*; вирівнювання — *по центру*.
5. Вставте до тієї ж клітинки зображення цуценяти, що міститься у файлі *Цуценя.jpg* у папці *Текстовий процесор*. Змініть його розмір, встановіть обтікання текстом *Навколо рамки,* щоб розмістити його за зразком (мал. 7.33).
6. У першому стовпці другого рядка введіть текст «тел. 050-0-66-66 Питати Марію», виділіть його й на вкладці *Макет* в групі *Вирівнювання* оберіть інструмент *Напрямок тексту*. Вирівняйте текст *по центру* й оберіть розмір символів — *10*.
7. Скопіюйте текст до інших стовпців другого рядка таблиці.
8. Збережіть документ з іменем *Оголошення* у власному форматі документів текстового процесора й у форматі PDF.



Мал. 7.33

9. Що таке недруковані знаки та як їх переглянути?

Деякі знаки визначають текстові об'єкти документа, але при друкуванні ніяк не відображаються. Це такі знаки, як пропуск, кінець абзацу, кінець клітинки таблиці тощо: за допомогою пропуска одне слово відділяється від іншого, при натисненні клавіші *Enter* створюється новий абзац, на перетині рядків і стовпчиків таблиці утворюється клітинка.

Іноді необхідно здійснити перехід на новий рядок раніше, без створення нового абзацу. У цьому разі застосовують примусовий розрив рядка, який виконується натисненням комбінації клавіш *Shift* + *Enter*. Такий прийом часто застосовується при введенні текстів віршів чи пісень, коли бажано, щоб кожні 4 рядки (або інша кількість рядків, залежно від необхідних для цього тексту правил оформлення) складали один абзац. Для позначення переходу на новий рядок також використовується недрукований символ.

При підготовці документа можна увімкнути режим, у якому недруковані знаки будуть відображені. Це дасть змогу не лише побачити об'єкти документа, а й відслідкувати деякі можливі помилки, щоб потім їх виправити. Адже в тексті, наприклад, між словами міг бути введений не один пропуск, а кілька, чи крапка в кінці речення була відірвана від останньої написаної літери тощо.

Для увімкнення режиму відображення недрукованих знаків використовується інструмент *Недруковані знаки* ¶ ( ), що розташований на вкладці *Основне* в групі *Абзац* у *Microsoft Word 2010* (на *Стандартній* панелі інструментів у *LibreOffice Writer*).

У режимі відображення недрукованих знаків можуть бути відображені різні знаки (табл. 7.1).

Таблиця 7.1

Недрукований знак	Опис
·	Пропуск
°	Нерозривний пропуск
¶	Кінець абзацу
↵	Перехід на новий рядок, не створюючи нового абзацу
¤	Кінець клітинки таблиці

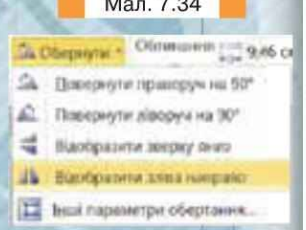
ДІЄМО

Вправа 6. Зміна вигляду документа в режимі відображення недрукованих знаків.

Мал. 7.34

Завдання. Перегляньте документ *Два кольори*, що зберігається в папці *Текстовий процесор*, у режимі відображення недрукованих знаків та визначте, які клавіші чи комбінації клавіш використовувалися для завершення кожного рядка. Додайте після назви вірша зображення, що збережене у файлі *Вишивка.jpg*, обітніть його та відобразьте зліва направо, щоб документ мав вигляд, як на зразку (мал. 7.34).

1. Відкрийте документ *Два кольори,* що зберігається в папці *Текстовий процесор*.
2. Увімкніть режим відображення недрукованих знаків за допомогою інструмента *Недруковані знаки* ¶ .
3. Визначте, натисненням якої клавіші чи комбінації клавіш завершувалося введення кожного з рядків вірша Д. Павличка «Два кольори». Скільки абзаців містить документ?
4. Додайте після назви вірша новий абзац, вставте до нього зображення, що зберігається у файлі *Вишивка.jpg* у папці *Текстовий процесор*.
5. Виділіть зображення, на вкладці *Формат* оберіть інструмент *Обтинання*. Наведіть вказівник миші на маркер обтинання в нижній частині зображення та виконайте протягування, щоб отримати фрагмент зображення, як на малюнку 7.34.
6. На вкладці *Формат* оберіть інструмент *Обернути* та вказівку *Відобразити зліва направо* (мал. 7.35).
7. Збережіть файл *Два кольори* у папці *Тексти* власної структури папок.



Мал. 7.35

ОБГОВОРЮЄМО

Обговоріть запитання, що містяться у файлі *Тема 7*, який зберігається у папці *Обговорюємо*. Також відкрити ці запитання можна за допомогою QR-коду.

ПРАЦЮЄМО В ПАРАХ

1. Порівняйте значки файлів, що збережені в різних форматах (мал. 7.1). Обговоріть у парах гіпотези щодо причин такого зображення.
2. Порівняйте параметри зміни нумерованого та маркованого списку (мал. 7.8, 7.9), знайдіть спільне й відмінне. Обговоріть у парах.
3. Наведіть приклади використання колонок у текстових документах. Обговоріть, яка кількість колонок є доцільною за яких умов.
4. Чим відрізняються спеціальні символи та недруковані знаки? Коли вони використовуються? Обговоріть у парах.
5. Пограйте в гру «Формула» — по черзі наведіть приклади математичних, хімічних та фізичних формул, які не можна ввести з клавіатури, а слід застосувати засоби створення формул.
6. Чи можна перемістити до іншого місця вставлену в документ таблицю? Як це зробити? Чи можна перемістити частину таблиці? Обговоріть у парах.

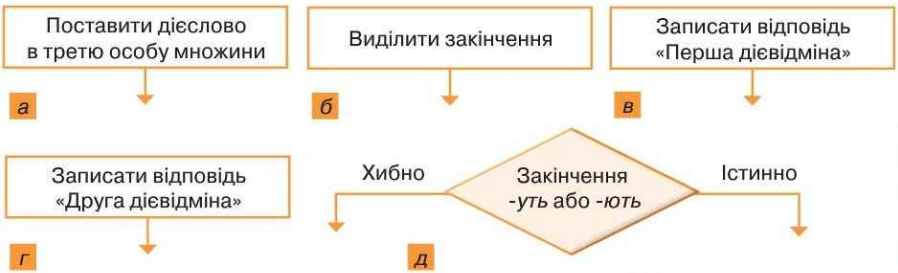
ЦІКАВІ ДЖЕРЕЛА

Робота з об'єктами в *Microsoft Word 2010*: *http://e-helper.com.ua/msword_lections*

7. Чи можна вміст окремої клітинки таблиці відформатувати інакше, ніж інші частини таблиці? Обговоріть у парах.
8. Чи можна визначити кращий спосіб створення таблиці засобами текстового процесора? Наведіть приклади ситуацій, коли доцільно надати перевагу конкретному способу. Обговоріть у парах.

ПРАЦЮЄМО САМОСТІЙНО

1. У папці *Текстовий процесор/Формат* розміщені текстові файли, що збережені в різних форматах. Вони містять однакові відомості. Порівняйте формат й обсяг кожного файла. Побудуйте для порівняння гістограму в середовищі табличного процесора, зробіть висновок. Завантажте в середовищі текстового процесора кожний із файлів, перегляньте та зробіть висновок.
2. Відкрийте файл *Про формати,* що збережений у папці *Текстовий процесор,* і розмістіть текст у дві колонки. Збережіть цей файл до папки *Тексти* власної структури папок у форматах *DOC, DOCX, ODT, RTF* та *PDF*. Визначте обсяг файла в різних форматах. У якому з п'яти форматів обсяг файла найменший?
3. Створіть у середовищі текстового процесора блок-схему алгоритму визначення дієвідміни дієслова. Скористайтеся засобами для створення фігур. В алгоритмі використайте команди, подані на малюнку 7.36. Розпочніть із команди, зображеної на малюнку 7.36, *а*.



Мал. 7.36

4. Створіть документ *Календар*, у якому за допомогою інструментів креслення та встановлення властивостей таблиць у текстовому документі оформіть у вигляді таблиці календар на поточний місяць і внесіть до відповідних клітинок дні народження друзів і найбільш важливі події на місяць. Наприклад, шаблон календаря на вересень 2016 р. показано на малюнку 7.37.
Розміри всіх клітинок зробіть однаковими; застосуйте до таблиці один зі стилів оформлення на власний смак, додайте до клітинок таблиці зображення із колекції кліпів, встановленої на вашому комп'ютері, для позначення подій.
5. Подайте відомості, що збережені у файлі *Винаходи українців* у папці *Текстовий процесор,* у вигляді таблиці з п'яти рядків та п'яти стовпчиків такої структури:

№	Винахід	Рік відкриття	Автор, портрет	Зображення винаходу

Вересень 2016 р.	Понеділок	Вівторок	Середа	Четвер	П'ятниця	Субота	Неділя
			1	2	3	4	5
	6	7	8	9	10	11	12
	13	14	15	16	17	18	19
	20	21	22	23	24	25	26
	27	28	29	30			

Мал. 7.37

Для введення даних до таблиці скористайтеся буфером обміну для переміщення необхідних відомостей. Доберіть в Інтернеті та додайте зображення вказаних винаходів. За потреби змініть розміри зображень або обітніть їх.

7. Створіть текстовий документ, що містить такі формули:

1) $(x_2 - x_1)^2 \geq 0$

2) $y = \dfrac{1}{x - \dfrac{1}{x}}$

3) $\begin{cases} 2x + 5y = 13, \\ 3x - 5y = -3 \end{cases}$

4) $Cu(OH)_2 \xrightarrow{t} CuO + H_2O$

5) $\rho = \dfrac{m}{V}$

ДОСЛІДЖУЄМО

Дослідіть, як у текстовому документі таблиця може розміщуватися відносно тексту. Для цього з контекстного меню таблиці оберіть вказівку *Властивості таблиці* й у вікні, що відкриється, визначте можливі значення для властивостей *Вирівнювання* та *Обтікання текстом* на вкладці *Таблиця*.

8. ПРАКТИЧНА РОБОТА 4

СТВОРЕННЯ ТЕКСТОВОГО ДОКУМЕНТА, ЩО МІСТИТЬ ОБ'ЄКТИ РІЗНИХ ТИПІВ

ПРИГАДАЙТЕ

- як створювати текстовий документ;
- як у текстовому документі створювати списки та колонки;
- як додавати до текстового документа різні об'єкти: символи, зображення, таблиці, формули;
- як форматувати об'єкти текстового документа.

СТВОРІТЬ

У власній структурі папок створіть папку *Практична робота 4*.

ПАМ'ЯТАЙТЕ

Під час виконання практичних завдань пам'ятайте про правила безпеки життєдіяльності при роботі з комп'ютером!

Завдання 1. Процесор (*8 балів*)

У документі *Процесор*, що зберігається в папці *Текстовий процесор*, перші три абзаци тексту розмістіть у дві колонки та додайте зображення за зразком. Для списку характеристик процесора змініть маркер на ▪ (мал. 8.1):

Процесор називають електронним "мозком" комп'ютера. Він призначений для автоматичного опрацювання та перетворення даних за наперед заданими програмами і управління роботою всіх пристроїв комп'ютера. Від його обчислювальної потужності істотно залежить продуктивність комп'ютера.

Процесор — це мікросхема, яка створюється на напівпровідниковому кристалі (або кількох кристалах) шляхом застосування складної мікроелектронної технології. Різноманітні операції в процесорі виконуються за спеціальними вказівками. Вказівки для процесора записують у комп'ютерній програмі.

Під час роботи процесор досить сильно нагрівається, тому на нього встановлюють систему охолодження — вентилятор, який називають *кулером*.

Основними характеристиками процесорів є:

- *тип* — відповідно до фірми-виробника розрізняють процесори Intel (Pentium, Celeron, Core2 Duo тощо), AMD (AMD64, Duron, Athlon тощо) та інших;
- *тактова частота* — визначає кількість виконуваних елементарних операцій за одну секунду, тобто швидкодію процесора; тактова частота сучасних процесорів вимірюється у гігагерцах (ГГц), уже розроблено процесори з частотою понад 3 ГГц;
- *розрядність* — максимальна довжина двійкового коду, який може опрацьовуватись або передаватись процесором; чим вища розрядність, тим вища потужність процесора;
- *кеш-пам'ять* — це внутрішня пам'ять процесора, яка дає можливість зберігати проміжні дані.

Мал. 8.1

Завдання 2. Раціональні вирази (*8 балів*)

Створіть новий документ *Вирази*, в якому за допомогою засобів створення формул і нумерованих списків введіть текст завдання:

Завдання.

Якому з наведених виразів тотожно дорівнює дріб $\frac{6x^2}{24x}$?

1) $\frac{x^2}{4}$; 2) $\frac{12x^3}{48x}$; 3) $\frac{x}{4}$; 4) $\frac{3x^4}{24x}$.

Завдання 3. Лабіринт (7 балів)

Створіть новий документ *Лабіринт*, у якому за допомогою засобів створення таблиць і їх обрамлення побудуйте лабіринт за зразком. Додайте та відформатуйте зображення, що збережене у файлі *Мишка.png* в папці *Текстовий процесор* (мал. 8.2).

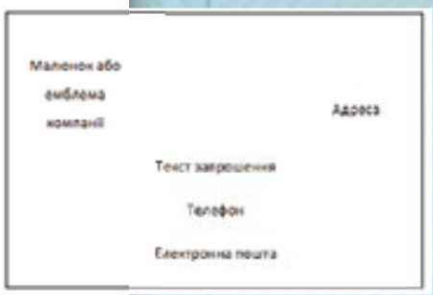
Мал. 8.2

Завдання 4. Сім чудес України (*8 балів*)

Доповніть і відформатуйте документ *Сім чудес України*, що зберігається в папці *Текстовий процесор.*

Завдання 5. Запрошення (*10 балів*)

Створіть новий документ *Запрошення*, у якому підготуйте запрошення на виставку «Я тобою, Україно, живу...». Застосуйте таблицю без оформлення для форматування даних за зразком (мал. 8.3).



Мал. 8.3

9. ОПРАЦЮВАННЯ СКЛАДНИХ ТЕКСТОВИХ ДОКУМЕНТІВ

ПРИГАДАЙТЕ:

- які параметри форматування символів та абзаців використовують під час оформлення тексту;
- як додати нумерацію сторінок;
- як надрукувати текстовий документ;
- як обрати режим роботи з документом;
- які дії виконують із документом у режимі розмітки документа та в режимі читання.

ВИ ДІЗНАЄТЕСЯ:

- як застосувати стильове форматування для оформлення документа;
- як у документі створити кілька розділів та відформатувати їх;
- що таке колонтитули та як їх додати до текстового документа;
- як можна переглянути структуру документа;
- як автоматично створити зміст і покажчик у текстовому документі;
- як створити текстовий документ на основі шаблону;
- як опрацьовувати складні текстові документи.

ВИВЧАЄМО

1. Як застосувати стильове форматування для оформлення документа?

Тексту можна надавати різного зовнішнього вигляду, змінюючи значення параметрів форматування символів, абзаців і сторінок документа. Ви вже вмієте змінювати шрифт, колір і розмір символів, їх накреслення, вирівнювання абзаців, відступи абзацу, міжрядковий інтервал та інші параметри форматування символів й абзаців. Полегшити та прискорити процес форматування тексту в документі можна за допомогою стилів.

Стилем називається сукупність параметрів форматування, що позначається унікальним іменем і може застосовуватися для форматування текстових фрагментів.

Стиль можна розглядати як вказівку форматування, створену користувачем. Користувач може обрати значення параметрів форматування, які потрібно застосувати (наприклад, шрифт і його розмір, вирівнювання, відстань між символами, відстань між абзацами, тип обрамлення тощо), об'єднати їх та надати унікальне ім'я цьому набору значень параметрів. Потім значення зазначених параметрів можна одночасно застосувати до виділених текстових фрагментів.

Для форматування тексту можна застосувати стилі двох видів: стиль символу та стиль абзацу.

Стиль символу змінює зовнішній вигляд окремих символів, слів, фраз. Для налаштування відповідних параметрів застосовуються інструменти для форматування символів та параметри у вікні *Шрифт*: шрифт і його розмір, а також вид накреслення, ефекти (закреслений, верхній індекс, контур) тощо.

Цікаво

Стиль (у загальному розумінні) — характерний вид, різновид чого-небудь, який виражається в деяких особливих ознаках, властивостях художнього оформлення.

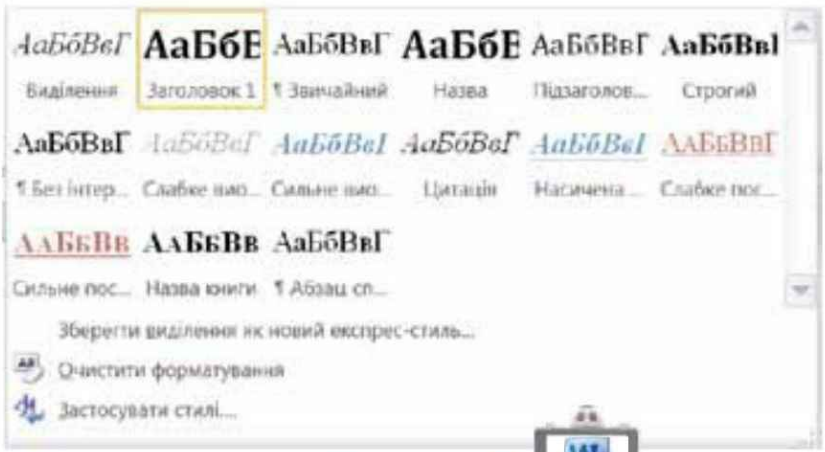
Стиль абзацу дає можливість змінити зовнішній вигляд абзацу: шрифт і розмір символів та інші атрибути тексту, а також міжрядковий інтервал, вирівнювання тексту, обрамлення та значення інших параметрів, які впливають на форматування всього абзацу.

Список *Стиль*, що в *Microsoft Word 2010* міститься на вкладці *Основне* та зображений на малюнку 9.1, *а* (у *LibreOffice Writer* — на панелі інструментів *Форматування*, мал. 9.1, *б*), містить перелік доступних стилів при роботі з поточним документом. Під час створення нового документа в цьому списку автоматично відображаються стандартні вбудовані стилі. Потім, при створенні користувачем нових стилів або зміні наявних, вони додаються до списку та в подальшому зберігаються разом із документом.

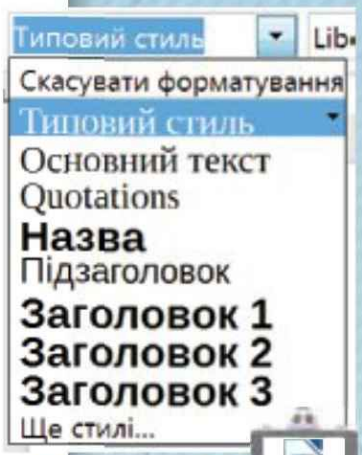
До стандартного набору стилів належать такі стилі: *Звичайний* (*Основний текст*), значення параметрів форматування якого застосовуються до введеного тексту, якщо не було вказано інших значень, *Заголовок 1*, *Заголовок 2*, ... , *Заголовок 9*, за допомогою яких можна формувати багаторівневу структуру великих документів тощо.

Щоб застосувати стиль до введеного тексту, достатньо виділити текстовий фрагмент, який потрібно відформатувати, й у списку *Стиль* обрати потрібний. Якщо назви потрібних стилів не відображаються у списку *Стиль*, то можна відкрити бічну панель *Стилі* (*Стилі та форматування*), яка містить повний набір вбудованих стилів, а також інструменти для внесення змін до існуючих стилів і створення нових (мал. 9.2, *а*, *б*). Щоб відкрити панель *Стилі*, у *Microsoft Word 2010* на вкладці *Основне* в групі *Стилі* слід натиснути кнопку , розташовану в правому нижньому куті групи. У *LibreOffice Writer* панель *Стилі та форматування* (мал. 9.2, *б*) з'являється, якщо обрати вказівку *Ще стилі* наприкінці списку стилів (мал. 9.1, *б*).

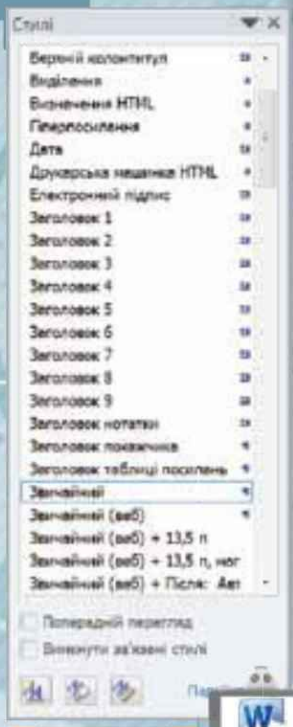
Різні стилі в архітектурі



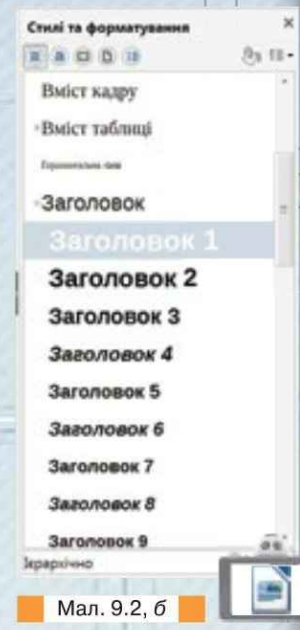
Мал. 9.1, *а*



Мал. 9.1, *б*



Мал. 9.2, *а*



Мал. 9.2, *б*

ДІЄМО

Вправа 1. Форматування документа з використанням стилів *Заголовки*.
Завдання. У документі *Кореспонденція*, що зберігається в папці *Текстовий процесор*, застосуйте до назви документа стиль *Заголовок 1*, до підзаголовків — стиль *Заголовок 3*.

1. Відкрийте документ *Кореспонденція*, що зберігається в папці *Текстовий процесор*.
2. Виділіть абзац, що містить назву документа. На вкладці *Основне* (на панелі інструментів *Форматування)* відкрийте список *Стиль* і виберіть стиль *Заголовок 1*.
3. Аналогічно застосуйте до підзаголовків у документі стиль *Заголовок 3*. Якщо назва стилю *Заголовок 3* не відображається в списку *Стиль*, відкрийте область *Стилі* (*Стилі та форматування*) (мал. 9.2, *а*, *б*).
4. Знайдіть в основному тексті слова, взяті в лапки. Послідовно виділяйте кожне з таких слів чи словосполучень та застосуйте до них стиль *Виділення*.
5. Збережіть результати у файлі з тим самим іменем у папці *Тексти* власної структури папок.

2. Як у документі створити кілька розділів та відформатувати їх?

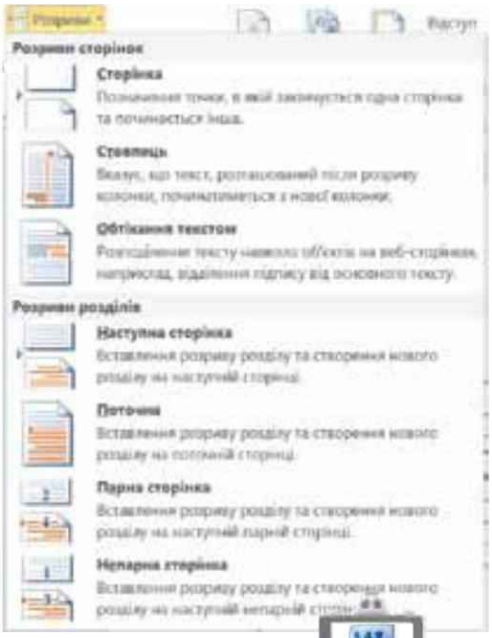
За замовчуванням текстовий документ складається з одного розділу. Усі сторінки одного розділу завжди мають однакові значення параметрів форматування сторінки. Однак часто доводиться опрацьовувати великі документи, які містять декілька розділів. Наприклад, електронний примірник книги чи підручника, збірка правил чи інструкцій тощо.

Як правило, новий розділ створюється, якщо потрібно для різних сторінок документа задати різні значення параметрів сторінки або щоб певна частина документа починалася з нової сторінки. Наприклад, новий розділ у романі прийнято розпочинати з нової сторінки. При розбитті тексту на колонки автоматично створюються розділи, оскільки кількість колонок — це одна з властивостей розділу.

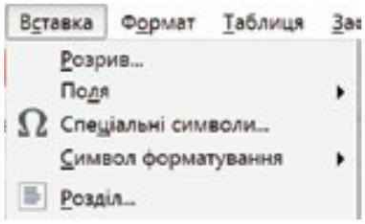
Розділ — частина документа, яка має задані значення параметрів форматування сторінки чи колонок.

Щоб розбити документ на декілька розділів, слід вставити розриви розділів, а потім задати значення параметрів форматування для кожного розділу.

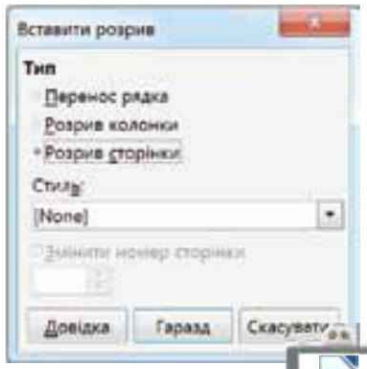
У *Microsoft Word 2010* для створення нового розділу в документі потрібно встановити текстовий курсор до того місця в документі, де потрібно вставити розрив розділу, і на вкладці *Розмітка сторінки* в групі *Параметри сторінки* обрати інструмент *Розриви*. У списку, що розкривається, в області *Розриви розділів* слід вибрати параметр, який вказує, звідки слід почати новий розділ (мал. 9.3). При цьому в документ вставляється мітка ~~Розрив розділу (з наступної сторінки)~~, яку можна побачити в режимі перегляду недрукованих знаків. Ця мітка позначає кінець розділу й містить відомості про такі параметри його форматування, як розміри полів, орієнтація сторінки, послідовність номерів сторінок тощо.



Мал. 9.3



Мал. 9.4, *а*



Мал. 9.4, *б*

Для створення нового розділу в *LibreOffice Writer* використовують вказівку *Вставка/Розділ* (мал. 9.4, *а*). Якщо потрібно, щоб розділ розпочинався з нової сторінки, виконують вказівку *Вставка/Розрив* й у вікні *Вставити розрив* обирають тип *Розрив сторінки* (мал. 9.4, *б*).

У текстовому документі або його розділі можна змінювати значення параметрів форматування сторінок, зокрема поля, орієнтацію сторінки, розмір паперу тощо.

Поля сторінки визначають відстань від краю аркуша паперу до початку відображення тексту.

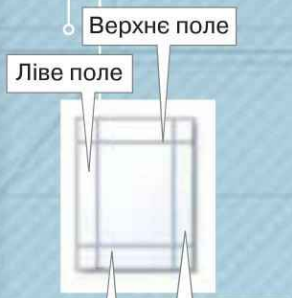
За замовчуванням документ друкується з одного боку аркуша паперу. У цьому разі розрізняють **верхнє**, **нижнє**, **ліве** та **праве** поля (мал. 9.5). Під час підготовки журнальних і книжкових видань застосовують двобічний друк. При цьому замість лівого та правого полів використовують поняття **внутрішніх** і **зовнішніх** полів (мал. 9.6).

Якщо передбачається зшивання надрукованого документа, то можна вибрати **розташування корінця** для переплетення — зліва чи зверху, та задати значення додаткової відстані.

Добираючи **розмір паперу**, слід враховувати, на якому принтері документ буде надруковано. Так, аркуш формату А4 можна роздрукувати на будь-якому сучасному принтері, тоді як формат паперу А3 підтримується лише спеціальними принтерами.

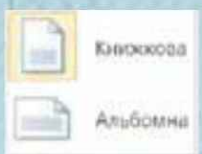
Орієнтація сторінки може бути **книжкова** — якщо сторінка розташовується вертикально, або **альбомна** — якщо горизонтально (мал. 9.7).

Щоб змінити значення параметрів форматування сторінок для певного розділу в *Microsoft Word 2010*, слід встановити текстовий курсор у межах

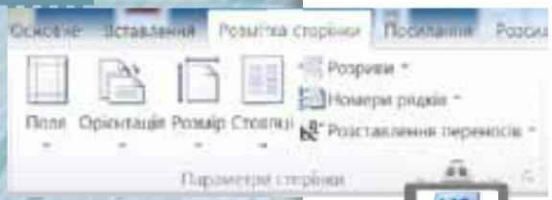


Мал. 9.5

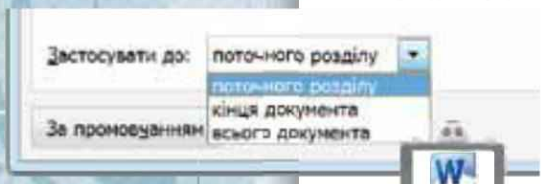
Мал. 9.6



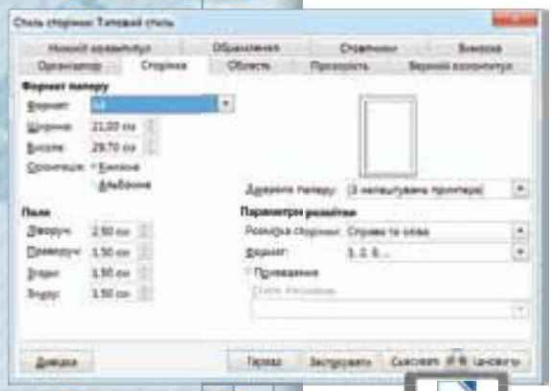
Мал. 9.7



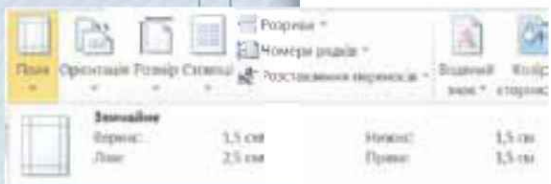
Мал. 9.8



Мал. 9.9



Мал. 9.10



Мал. 9.11

цього розділу, на вкладці *Розмітка сторінки* в області *Параметри сторінки* обрати відповідні інструменти (мал. 9.8) або натиснути кнопку , щоб викликати вікно *Параметри сторінки*, на відповідних вкладках якого можна вказати потрібні значення параметрів. Щоб вказані зміни впливали не на весь документ, а лише на обраний розділ, слід на будь-якій із вкладок *Поля, Папір, Макет*, на якій змінювалися значення параметрів, у списку *Застосувати до* вибрати значення *поточного розділу* (мал. 9.9).

Переглянути та змінити значення параметрів форматування сторінки в *LibreOffice Writer* можна у вікні *Стиль сторінки* (мал. 9.10), яке викликають за допомогою вказівки *Формат/Сторінка*. Розміри полів, орієнтацію сторінки та розмір паперу можна задавати на вкладці *Сторінка* цього вікна.

ДІЄМО

Вправа 2. Створення розділів у документі.

Завдання. Створіть у документі *Кореспонденція*, що зберігається в папці *Тексти* власної структури папок, розділи так, щоб кожний пункт документа містився в окремому розділі та розпочинався з нової сторінки. Встановіть орієнтацію сторінки для першого розділу — альбомну, а для всіх розділів документа значення розмірів полів: ліве — 2,5 см, решта — 1,5 см.

1. Відкрийте документ *Кореспонденція*, що зберігається в папці *Тексти* у вашій структурі папок.
2. Встановіть значення параметрів полів сторінок для всього документа: ліве — 2,5 см, решта — 1,5 см, застосувавши шаблон полів — *Звичайне* (мал. 9.11).
3. Встановіть текстовий курсор на початку абзацу, що містить підзаголовок *1. Про листи взагалі*. На вкладці *Розмітка сторінки* в групі *Параметри сторінки* оберіть інструмент *Розриви.* У списку, що розкривається, в області *Розриви розділів* оберіть *Наступна сторінка*.
4. Діючи аналогічно, вставте розділи до кожного пункту документа.
5. Встановіть текстовий курсор у межах першого розділу. На вкладці *Розмітка сторінки* в області *Параметри сторінки* оберіть інструмент *Орієнтація* та задайте альбомну орієнтацію сторінки.
6. Перегляньте документ. Зробіть висновки, чи змінилась орієнтація сторінки на альбомну для всіх сторінок документа.
7. Збережіть результати у файлі з тим самим іменем.

3. Що таке колонтитули та як їх додати до текстового документа?

Важливими параметрами сторінки є наявність та налаштування **колонтитулів**, у яких можна вказувати відомості про назву документа чи окремого розділу, дату створення, автора, номер сторінки, адреси інтер-

Мал. 9.12

нет-ресурсів, які є джерелом матеріалів тощо. Колонтитули часто можна побачити в книжках і журналах. Наприклад, у книжках на парних сторінках може бути вказано автора книги, а на непарних — її назву. У журналах в одному з колонтитулів зазвичай вказують назву та номер журналу, а також рік видання. Номери сторінок, які ви вже вмієте додавати до документа, розташовуються саме в області верхнього або нижнього колонтитула документа (мал. 9.12).

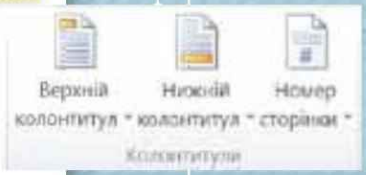
Колонтитул — об'єкт, що розміщується над текстом (верхній колонтитул) або під текстом (нижній колонтитул) кожної сторінки книжки, газети, журналу, документа.

Цікаво

Назва **колонтитул** утворюється з двох слів: від франц. *colonne* — стовпчик і лат. *titulus* — напис, заголовок.

Колонтитули можуть містити текст і деякі інші об'єкти: малюнки, лінії або інші фігури тощо.

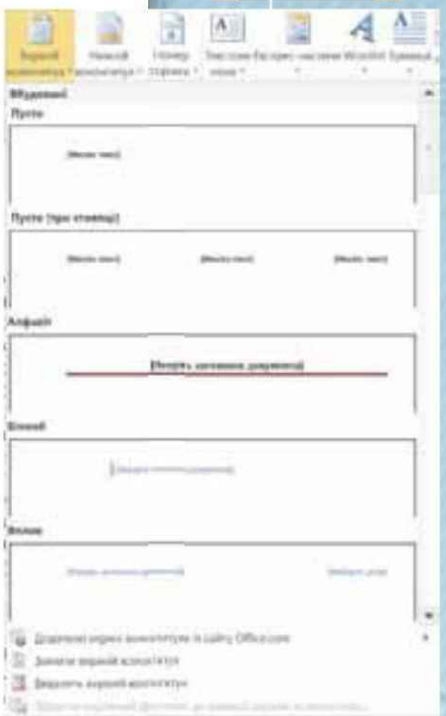
Якщо в документі є кілька розділів, то колонтитули можуть бути однакові для всіх розділів, або кожний розділ може містити різні колонтитули.



Мал. 9.13

У *Microsoft Word* вводити та редагувати вміст колонтитулів можна в окремих областях у верхній чи нижній частині сторінки. Щоб перейти до області колонтитула, використовують інструменти з вкладки *Вставлення* в групі *Колонтитули* (мал. 9.13). Інструмент *Номер сторінки* використовують для додавання та налаштування в області колонтитула номерів сторінки. Особливо це зручно, якщо колонтитул не містить інших об'єктів.

Список, що розкривається при виборі інструментів *Верхній колонтитул* чи *Нижній колонтитул*, містить бібліотеку вбудованих стилів колонтитулів (мал. 9.14), з яких можна обрати той, що більше підходить за оформленням чи наявністю необхідних об'єктів.



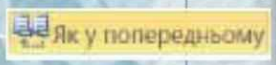
Мал. 9.14

Після вибору одного з вбудованих стилів колонтитулів текстовий курсор автоматично встановлюється в крайній лівій частині області колонтитула. Якщо певні об'єкти потрібно розмістити по центру колонтитула або в правій його частині, швидко перевести текстовий курсор у потрібне положення можна за допомогою клавіші *Tab*. Під час роботи з колонтитулами текст документа відображається неяскравим кольором і на стрічці в області *Знаряддя для колонтитулів* з'являється вкладка *Конструктор*, що містить інструменти для роботи з колонтитулами (мал. 9.15).

Щоб завершити створення чи редагування колонтитулів і повернутися до вмісту документа, слід на вкладці *Конструктор* натиснути кнопку *Закрити колонтитули* або двічі клацнути мишею за межами області колонтитулів.

Мал. 9.15

Якщо документ складається з кількох розділів, за замовчуванням увімкнено режим *Як у попередньому*, і колонтитули всіх розділів будуть однаковими. Якщо необхідно створити різні колонтитули в кожному розділі, то після створення колонтитула першого розділу слід перейти до колонтитула наступного розділу та на вкладці *Конструктор* обрати інструмент *Як у попередньому*, щоб вимкнути цей режим. Після цього можна створювати новий колонтитул в обраному розділі.



Якщо документ містить нумерацію сторінок, за замовчуванням усі сторінки нумеруються послідовно. У межах розділів можна налаштувати інші параметри нумерації: вказати, з якого номера починати нумерувати сторінки розділу, долучити до номера сторінки номер розділу, наприклад, 1-1, 1-2, 1-3 та 2-1, 2-2 тощо.

У *LibreOffice Writer* додати колонтитули до документа можна за допомогою вказівок *Верхній колонтитул* і *Нижній колонтитул* з меню *Вставка*. При цьому до тексту в межах колонтитула застосовується типовий стиль колонтитулів.

ДІЄМО

Вправа 3. Створення колонтитула в документі.

Завдання. У документ *Кореспонденція*, що зберігається в папці *Тексти* власної структури папок, вставте нижній колонтитул, у якому відображатиметься назва документа, поточна дата та номер сторінки.

1. Відкрийте документ *Кореспонденція*, що зберігається в папці *Тексти* у вашій структурі папок.
2. На вкладці *Вставлення* в групі *Колонтитули* оберіть інструмент *Нижній колонтитул* (мал. 9.13) та шаблон *Алфавіт*.
3. У лівій частині нижнього колонтитула введіть із клавіатури назву документа *Кореспонденція*. На вкладці *Конструктор* в області *Знаряддя для колонтитулів* (мал. 9.15) оберіть інструмент *Дата та час*, у вікні *Дата й час* оберіть такий формат дати, щоб місяць було подано числом, наприклад *3.11.2016*, та натисніть кнопку *ОК*.
4. Натисніть клавішу *Tab*, щоб перейти до правої межі області колонтитула. На вкладці *Конструктор* в області *Знаряддя для колонтитулів* оберіть інструмент *Номер сторінки*.
5. Виконайте затримку миші над інструментами вкладки *Конструктор* в області *Знаряддя для колонтитулів* та визначте їх призначення.
6. На вкладці *Конструктор* оберіть інструмент *Закрити колонтитули*.
7. Переконайтеся, що створений колонтитул повторюється на всіх сторінках документа.
8. Збережіть результати у файлі з тим самим іменем.

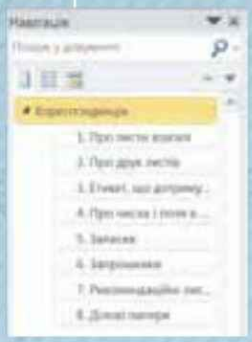
4. Як можна переглянути структуру документа?

Заголовки розділів і підрозділів документа визначають його **структуру**. Текстові процесори містять засоби, за допомогою яких можна переглядати структуру документа та використовувати її для швидкого переміщення в потрібне місце документа. Щоб скористатися такими засобами, необхідно, щоб для абзаців, які містять такі заголовки, параметр *Рівень контуру* у вікні *Абзац* мав значення, відмінне від *Основний текст* — *Рівень 1*, *Рівень 2* тощо (мал. 9.16). У вбудованих стилях заголовків номер рівня відповідає номеру в назві стиля: *Заголовок 1* має *Рівень 1*, *Заголовок 2* — *Рівень 2* і т.д. Тому для використання засобів роботи зі структурою можна застосувати до заголовків розділів і підрозділів відповідні стандартні стилі заголовків.

Мал. 9.16

Переглянути структуру документа та переміщуватися в місце документа, де розпочинається підрозділ з обраним заголовком, у *Microsoft Word 2010* можна за допомогою панелі *Навігація* (мал. 9.17), яка відображається на екрані, якщо на вкладці *Вигляд* у групі *Відображення* увімкнути прапорець *Область переходів* (мал. 9.18).

Опрацьовувати великі документи в *Microsoft Word 2010* можна також у *Режимі структури*, в якому можна згорнути документ, відображаючи лише основні заголовки, переміщувати чи копіювати частини тексту за допомогою заголовків тощо. У цьому режимі текст подано без урахування параметрів форматування абзаців. Усі заголовки та звичайний текст відображено з відступами, які представляють рівень цього тексту в загальній структурі документа (мал. 9.19). Ліворуч від заголовків відображаються позначки , клацання на яких приводить до виділення розділу, що відповідає обраному заголовку, а подвійне клацання — до згортання чи розгортання відображення підзаголовків обраного заголовка.



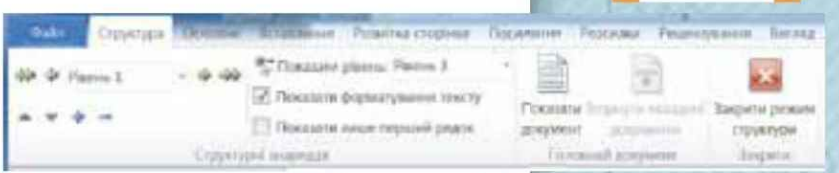
Мал. 9.17

Щоб перейти до режиму структури, слід на вкладці *Вигляд* у групі *Режими перегляду документа* обрати інструмент *Структура* або в правій частині рядка стану натиснути кнопку *Структура* .

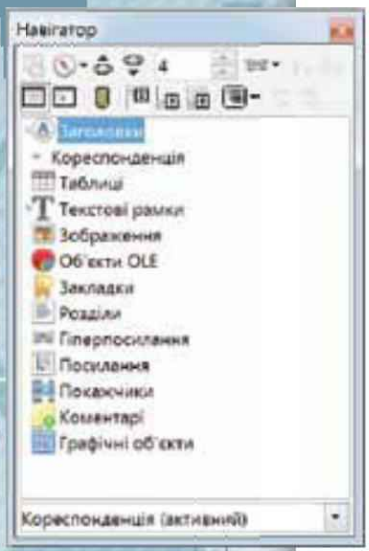
Мал. 9.18

Під час роботи в режимі структури на стрічці відкривається вкладка *Структура* (мал. 9.20), що в групі *Структурні знаряддя* містить інструменти, за допомогою яких можна обрати ступінь деталізації. Наприклад, відображати всі заголовки чи лише заголовки від першого до третього рівня, підвищити чи понизити рівень для певних заголовків, переміщувати підрозділи в тексті тощо.

Мал. 9.19



Мал. 9.20



Мал. 9.21

Мал. 9.22

Щоб повернутися в режим розмітки документа, слід на вкладці *Структура* обрати інструмент *Закрити режим структури*.

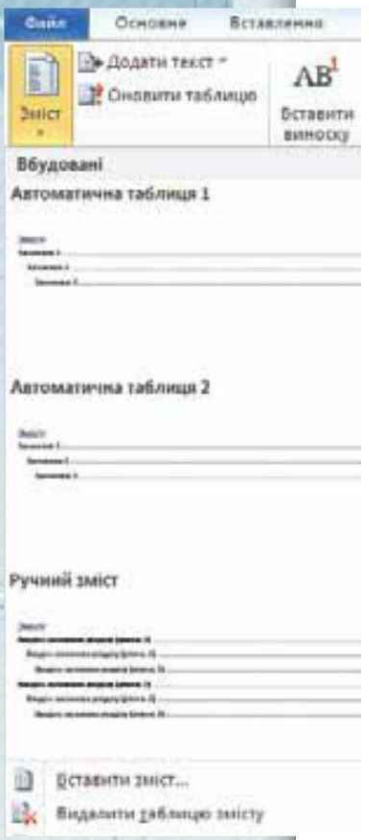
У *LibreOffice Writer* заголовки та інші об'єкти документа можна переглянути та використати для переміщення в документі на панелі *Навігатор* (мал. 9.21), яку можна відкрити за допомогою вказівки меню *Перегляд/Навігатор* або кнопки *Навігатор* на бічній панелі, що розташована в правій частині вікна (мал. 9.22).

Заголовки нижчих рівнів можуть бути приховані, — щоб їх відобразити, слід клацнути на позначці ⊞ зліва від заголовка першого рівня.

Вправа 4. Перегляд документа в *Режимі структури*.

Завдання. Перегляньте документ *Повість-казка* в режимі відображення структури. Не переглядаючи тексту твору, визначте, скільки розділів містить повість.

1. Відкрийте документ *Повість-казка*, що зберігається в папці *Текстовий процесор*.
2. Перейдіть до режиму *Структура документа*.
3. Оберіть другий рівень деталізації структури. Скористайтесь інструментом Показати рівень: Рівень 2 на вкладці *Структура* текстового процесора *Microsoft Word 2010* або позначкою ⊞ вікна *Навігатор* у *LibreOffice Writer*.
4. Полічіть кількість розділів документа. Дослідіть, як виглядатиме документ, якщо обрати третій рівень структури.
5. Збережіть зміни у файл *План повісті* в папку *Тексти* власної структури папок.
6. Відкрийте збережений файл. Перевірте, чи зберігається режим відображення структури після збереження файла, зробіть висновок. Закрийте вікно текстового процесора.



Мал. 9.23

5. Як автоматично створити зміст і покажчик у текстовому документі?

Великі документи, як правило, містять зміст та інші засоби для полегшення орієнтування. **Зміст** являє собою список заголовків документа та номерів сторінок, на яких починаються розділи з цими заголовками, і розміщується зазвичай на початку документа. Наприкінці великих документів, у яких трапляється багато важливих понять, створюють **покажчик** — словник з основних понять документа, які розташовані в алфавітному порядку з вказуванням номера сторінки, на якій це поняття міститься в документі. Наприклад, зміст і

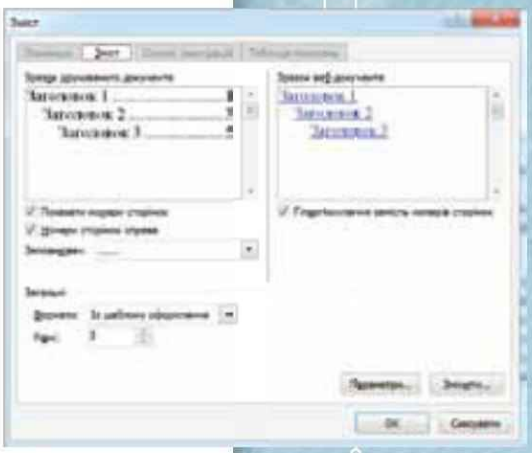
покажчик є в підручнику з інформатики, який ви тримаєте в руках.

Текстові процесори надають можливість автоматичного створення змісту та покажчика, які можна також використовувати для переміщення по документу, оскільки кожний елемент такого об'єкта є **посиланням**. Наприклад, якщо клацнути мишею при натисненій клавіші *Ctrl* на деякому заголовку в змісті, текстовий курсор переміститься в місце документа, де розпочинається частина документа із цим заголовком.

Мал. 9.24

Зміст можна створити автоматично, до нього будуть включені абзаци, у яких значення параметра *Рівень контуру* відрізняється від *Основний текст* або до яких застосовано стандартні стилі *Заголовок 1, Заголовок 2* тощо. Тому, перш ніж додавати зміст, необхідно сформувати структуру документа та встановити текстовий курсор на те місце в документі, де його потрібно вставити.

Щоб додати зміст у *Microsoft Office 2010*, слід на вкладці *Посилання* обрати інструмент *Зміст* (мал. 9.23).



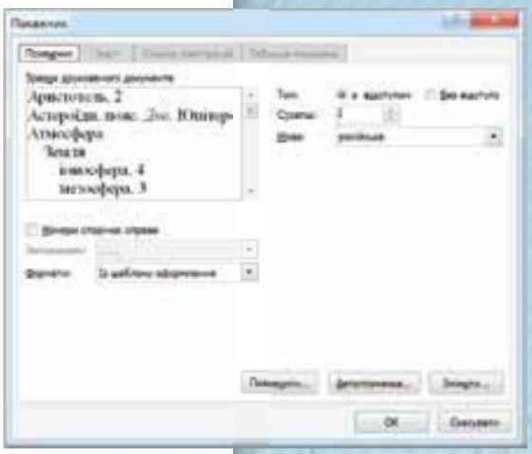
Мал. 9.25

Потрібний стиль можна обрати у списку, що розкривається. При цьому в документі автоматично будуть знайдені всі абзаци, до яких застосовані стилі заголовків, та буде сформовано зміст у документі (мал. 9.24). Якщо необхідно задати особливі налаштування змісту, слід обрати вказівку *Вставити зміст* (мал. 9.23). При цьому відкривається вікно *Зміст* (мал. 9.25), у якому можна обрати режим додавання номерів сторінок, із яких починається кожний із розділів і підрозділів, обрати формат оформлення, вказати кількість рівнів заголовків, які необхідно включати до змісту тощо.

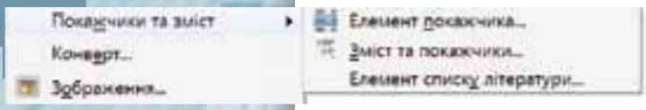
Мал. 9.26

Для створення покажчика в *Microsoft Office 2010* використовують інструменти із групи *Покажчик* на вкладці *Посилання* (мал. 9.26).

Спочатку необхідно в документі послідовно виділити поняття, які слід включити до покажчика, та обрати для кожного з них інструмент *Позначити елемент*. Коли всі елементи позначені, слід встановити текстовий курсор у те місце документа, де має бути створений покажчик, наприклад, наприкінці документа, та обрати інструмент *Покажчик*. У вікні *Покажчик* (мал. 9.27), що відкривається при цьому, можна змінити кількість стовпців (колонок), у які слід розмістити поняття в покажчику, та інші параметри.



Мал. 9.27



Мал. 9.28

Створення змісту та покажчика в *LibreOffice Writer* відбувається аналогічно. Для цього використовують вказівку меню *Вставка/Покажчики та зміст* (мал. 9.28).

Налаштування властивостей виконують у вікні *Вставити Покажчик/Зміст*.

ДІЄМО

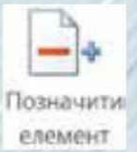
Вправа 5. Створення змісту документа.

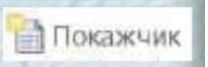
Завдання. Для документа *Кореспонденція*, що зберігається в папці *Тексти* власної структури папок, автоматично створіть зміст.

1. Відкрийте документ *Кореспонденція*, що зберігається в папці *Тексти* власної структури папок.
   У цьому документі до заголовків розділів і підрозділів у вправі 1 ви застосували стилі *Заголовок 1* та *Заголовок 3*.
2. На початку документа вставте новий абзац і встановіть до нього текстовий курсор.
3. На вкладці *Посилання* оберіть інструмент *Зміст* і в списку, що розкривається, оберіть один зі стилів оформлення змісту (мал. 9.23).
5. Створіть новий розділ, щоб основний текст документа розпочинався з наступної сторінки після змісту. Для цього на вкладці *Розмітка сторінки* в групі *Параметри сторінки* оберіть інструмент *Розриви*. У списку, що розкривається, в області *Розриви розділів* оберіть *Наступна сторінка*.
6. Поверніться до змісту. Натисніть на клавіатурі клавішу *Ctrl* і клацніть на назві заголовка *Етикет, якого дотримуються в листах* (мал. 9.24). Що при цьому відбувається?
7. Збережіть документ із тим самим іменем.

Вправа 5. Створення алфавітного покажчика.

Завдання. Для документа *Кореспонденція*, що зберігається в папці *Тексти* власної структури папок, автоматично створіть алфавітний покажчик понять, які використовуються в документі.





1. Відкрийте документ *Кореспонденція*, що зберігається в папці *Тексти* власної структури папок.
2. Знайдіть у тексті документа такі основні поняття: *кореспонденція, листи, звернення, приступ, розповідь, висновок, постскриптум, конверти, етикет, записки, запрошення, рекомендаційні листи, ділові папери*.
3. Послідовно виділяйте кожне з понять та обирайте інструмент *Позначити елемент* на вкладці *Посилання* в групі *Покажчик*.
4. Встановіть текстовий курсор у новому абзаці в кінці документа та оберіть інструмент *Покажчик* на вкладці *Посилання*.
5. У вікні *Покажчик* перегляньте властивості, встановлені за замовчуванням (мал. 9.27), та натисніть кнопку *ОК*.
6. Клацніть мишею на слові *етикет* у створеному покажчику. Переконайтеся, що на екрані відобразився фрагмент документа, що містить це поняття. Зробіть висновок про використання елементів покажчика.
7. Збережіть документ із тим самим іменем.

6. Як створити текстовий документ на основі шаблону?

Для полегшення створення типових документів — листів, візитівок, резюме, звітів тощо — використовують шаблони.

Шаблон документа — це текстовий документ, що містить усі елементи, які є «спільними» для різних документів даного типу. Шаблон використовується як зразок для створення нових документів певного типу.

Як правило, у шаблоні визначено стилі, які застосовано в цих документах; шаблони також можуть містити колонтитули, будь-який готовий текст, зображення тощо.

Під час створення нового документа за шаблоном користувач починає не «з чистого аркуша», а з копії шаблону. Наприклад, якщо існує готовий шаблон для наказів керівництва компанії, то під час створення нового наказу його заголовок, зображення логотипа тощо вже розміщені на своїх місцях, потрібно додати лише номер і текст наказу.

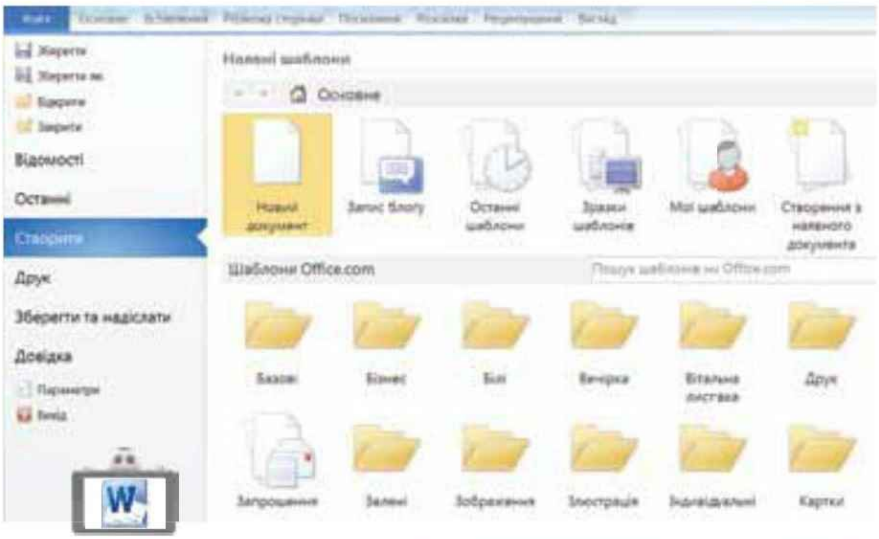
Шаблони часто використовують для того, щоб установити єдиний стандарт для типових документів в організації.

Шаблон — це звичайний текстовий документ, але він має інше розширення. Наприклад, у *Microsoft Word* файли документів мають розширення *doc* або *docx*, а шаблонів — *dot* або *dotx*. У *LibreOffice Writer* файли шаблонів мають розширення *ott*.

При створенні документа на основі шаблону відкривається не сам файл шаблону, а створюється новий документ, що містить усі об'єкти, як і у файлі шаблону. При внесенні змін до такого документа файл шаблону залишається незмінним, тому його можна використовувати багато разів. Текстові процесори також дають змогу відкривати й редагувати шаблони аналогічно до редагування документів. Можна перетворювати документ на шаблон. Основна відмінність між документами та шаблонами полягає в їх використанні.

Існує два основні типи шаблонів: глобальні й шаблони документа.

Будь-який документ створюється на основі шаблону. За замовчуванням нові документи створюються на основі глобального шаблону *Звичайний* (*Normal*), який містить параметри форматування, доступні для всіх



Мал. 9.29

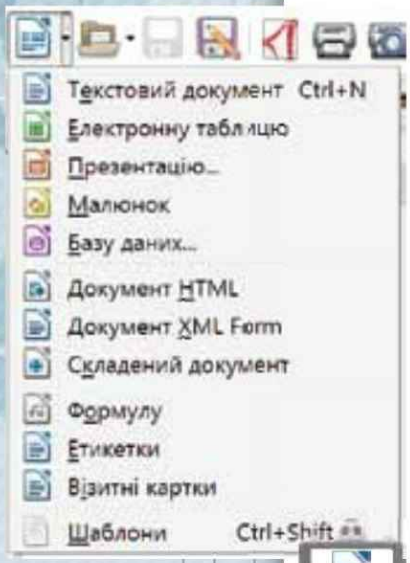
Цікаво

У розширенні файлів шаблонів одна з літер розширення текстового файла замінюється літерою *t* — від англ. *template* — шаблон.

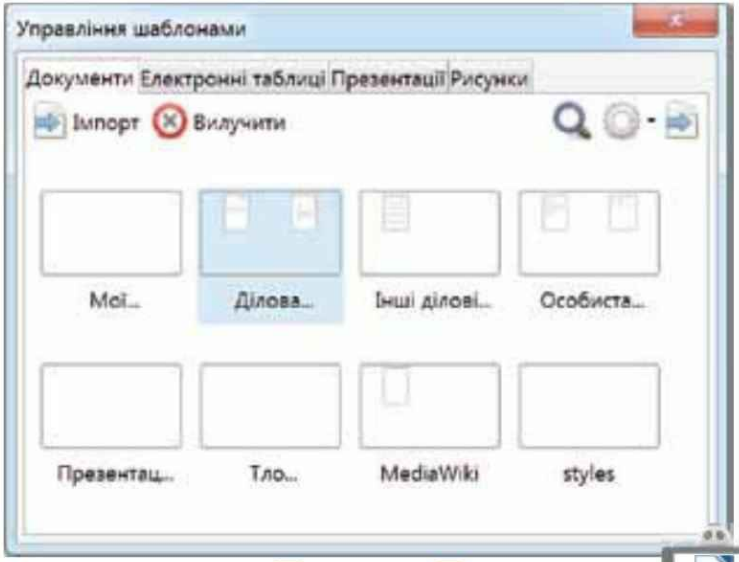
Цікаво

Якщо потрібно створити набір документів, які мають схоже форматування, слід використати шаблони з однієї групи: *Звичайні*, *Вишукані* або *Сучасні*. Так, якщо для створення резюме використовувався шаблон *Вишукане резюме*, то для підготовки листа до нього слід використати шаблон *Вишуканий лист*.

Мал. 9.30



Мал. 9.31



Мал. 9.32

документів. Шаблони документа, наприклад шаблони записки або резюме, містять параметри форматування, доступні тільки для документів, які були створені за допомогою цього шаблону.

Шаблони створюються на допомогу користувачеві, їх використання зберігає час на підготовку ділових паперів. Різні текстові процесори містять власні набори шаблонів.

Пакет програм *Microsoft Office* містить велику кількість різних шаблонів, деякі з текстових шаблонів можна відкрити безпосередньо в середовищі текстового процесора, іншими — шаблонами *Office.com* — можна скористатися за наявності підключення до Інтернету.

Для створення документа на основі шаблону документа в середовищі текстового процесора *Microsoft Word 2010* слід скористатися вказівкою *Файл/Створити*, в області *Наявні шаблони* обрати *Зразки шаблонів* (мал. 9.29) і серед запропонованих обрати потрібний шаблон документа. У правій частині екрана слід встановити перемикач у положення *Документ* і натиснути кнопку *Створити* (мал. 9.30). Далі необхідно ввести потрібний текст до кожного поля з відповідними підказками такого документа.

Документ можна створювати й на основі нестандартного шаблону, для цього потрібно лише відкрити шаблон як звичайний документ, а потім продовжувати працювати з його вмістом.

У текстовому процесорі *LibreOffice Writer* інструмент *Створити* на панелі інструментів *Стандартна* містить список, що розкривається, для створення стандартних документів пакета *LibreOffice* (мал. 9.31). Серед додаткових шаблонів текстових документів можна обрати *Формулу*, *Етикетки* та *Візитні картки*. Також можна створювати нові *Шаблони* з використанням інструментів вікна *Управління шаблонами* (мал. 9.32).

ДІЄМО

Вправа 6. Створення документа на основі шаблону.
Завдання. Створіть новий документ на основі шаблону *Анкета.dotx (Анкета.ott)*, що зберігається в папці *Текстовий процесор*.

1. Відкрийте папку *Текстовий процесор* і двічі клацніть на значку шаблону *Анкета.dotx* (*Анкета.ott*).
2. Перегляньте вміст нового документа. Визначте, використовуючи рядок заголовка вікна текстового процесора *Microsoft Word*, чи збережений новий документ і яке ім'я він має.
3. Користуючись запропонованим шаблоном, заповніть анкету «Який я?».
4. Збережіть документ з іменем *Анкета.docx* (*Анкета.odt*) у папці *Тексти* власної структури папок.

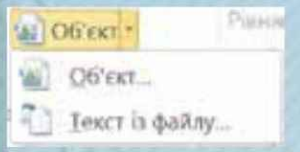
Вправа 7. Створення документа за допомогою вбудованого шаблону.
Завдання. Створіть файл *Лист* за допомогою відповідного стандартного шаблону текстового процесора.

1. Завантажте текстовий процесор *Microsoft Word 2010*. Виконайте вказівку *Файл/Створити*, в області *Наявні шаблони* оберіть *Зразки шаблонів*. Серед переліку шаблонів оберіть *Сучасний лист*, у правій частині вікна встановіть перемикач у положення *Документ* і натисніть кнопку *Створити*.
2. Введіть своє ім'я й замість текстових фрагментів, що записані у квадратних дужках, наприклад [*Введіть адресу відправника*], введіть власні дані.
3. Підготуйте текст листа з проханням про надання від авторів дозволу використовувати їхній твір, наприклад, для дослідницького проекту.
4. Видаліть запропоноване зображення та вставте на його місце власне фото.
5. Збережіть документ з іменем *Лист* у папці *Тексти* власної структури папок.

7. Як опрацьовувати складні текстові документи?

Ефективне опрацювання складного документа, що складається з кількох розділів і містить велику кількість сторінок, передбачає виконання певних дій:

1. Введення тексту та додавання об'єктів до документа.
2. Якщо необхідні текстові фрагменти збережені в іншому файлі, можна скористатися буфером обміну для їх копіювання та вставлення до документа. Об'єднувати текст декількох файлів можна за допомогою вказівки *Текст із файлу* зі списку інструмента *Об'єкт*, що розташований на вкладці *Вставлення* в групі *Текст* (мал. 9.33).
   Далі у вікні *Вставлення файлу* обирають потрібний файл і натискають кнопку *Вставити*.
3. Форматування тексту та об'єктів документа.
4. Форматування заголовків розділів і пунктів із використанням вбудо-



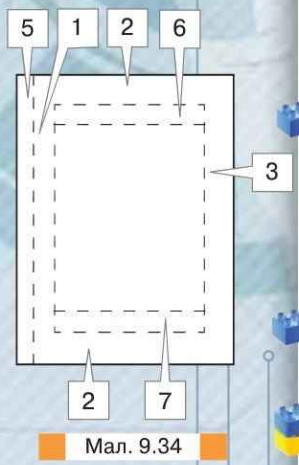
Мал. 9.33

ваних стандартних стилів заголовків або користувацьких параметрів форматування, що визначають рівень абзацу.

5. Створення в документі необхідної кількості розділів і налаштування параметрів сторінки для кожного розділу окремо чи для всього документа.
6. Перегляд структури документа за допомогою засобів навігації або в режимі структури, корекція послідовності розташування розділів у разі необхідності.
7. Додавання до документа засобів орієнтування та навігації: змісту, алфавітного покажчика тощо.

ОБГОВОРЮЄМО

Обговоріть запитання, що містяться у файлі *Тема 9*, який зберігається у папці *Обговорюємо*. Також відкрити ці запитання можна за допомогою QR-коду.



Мал. 9.34

ПРАЦЮЄМО В ПАРАХ

1. Зобразіть засобами текстового процесора модель документа, який містить вказані елементи за зразком (мал. 9.34).
   - верхній колонтитул;
   - нижній колонтитул;
   - ліве поле;
   - праве поле;
   - корінець;
   - верхнє поле;
   - нижнє поле.

   Запропонуйте в парі визначити їх розміщення на моделі.
2. Як внести деякі зміни до будь-якого наявного шаблону? Чи можна створити набір шаблонів, потрібних для учня в школі? Назвіть п'ять прикладів таких документів. Обговоріть у парах. Створіть спільний список.
3. Якщо створити власний шаблон і зберегти його у стандартній папці *Шаблони*, то чи буде такий шаблон відображатись у діалоговому вікні *Шаблони*, яке з'являється на екрані після вибору на панелі завдань посилання *На моєму комп'ютері?* Якщо так, то в якому розділі, з якою назвою? Обговоріть у парах.
4. Для чого у друковані видання вставляють колонтитули? Назвіть три переваги використання колонтитулів у друкованих документах. Чи можуть в одному виданні міститися різні колонтитули? Обговоріть у парах.
5. Обговоріть можливі причини, коли в документі доцільно використовувати зміст, а коли — режим структури.

ПРАЦЮЄМО САМОСТІЙНО

1. Створіть власну візитівку на основі відповідного вбудованого шаблону.
2. Відкрийте документ *Пам'ятники Києва*, що зберігається в папці *Текстовий процесор*. Скористайтеся схемою документа та визначте, скільки заголовків містить документ. Перегляньте, які стилі застосовано до заголовків документа. Встановіть такі параметри форматування сторінки: *ліве поле* — 3 см, *праве поле* — 1,5 см.
3. Відкрийте документ *Пам'ятники Києва*, що зберігається в папці *Текстовий процесор*. Визначте, які стилі застосовано до заголовків документа.

Створіть зміст на початку документа. Додайте до документа номери сторінок і нижній колонтитул, що містить текст *Пам'ятники Києва*.

4. Відкрийте документ *Структура ділового листа*, що зберігається в папці *Текстовий процесор*. Застосуйте до заголовків і підзаголовків документа стилі заголовків відповідних рівнів. Створіть зміст на початку документа. Додайте наприкінці документа матеріали з файла *Додатки,* що зберігається в папці *Текстовий процесор*. Скористайтеся буфером обміну або вказівкою *Текст із файлу*.
5. Відкрийте документ *Пам'ятники Києва*, що зберігається в папці *Текстовий процесор*. Розбийте документ на розділи таким чином, щоб відомості про кожний із пам'ятників починалися з нової сторінки. Створіть у документі колонтитули, в яких на непарних сторінках відображатиметься назва *Пам'ятники Києва*, на парних сторінках — назва пам'ятника, що стосується певного розділу. Щоб мати можливість у різних розділах створити різні колонтитули, відключіть режим *Як в попередньому розділі*, для цього оберіть відповідний інструмент на вкладці *Конструктор* в області *Знаряддя для колонтитулів*.
6. Створіть похвальну грамоту на основі шаблону *Грамота*, що зберігається в папці *Текстовий процесор*. На його основі створіть грамоту для конкретної людини за успішне опанування текстового процесора.
7. Оформіть реферат, який ви готували з іншого предмета, за алгоритмом опрацювання складного документа. Передбачте використання стилів заголовків, розбиття документа на розділи, налаштування колонтитулів та параметрів сторінки документа, автоматичне додавання змісту на початку документа.

ДОСЛІДЖУЄМО

Дослідіть, які параметри на вкладці *Макет* у вікні *Параметри сторінки* в *Microsoft Word 2010* (на вкладках *Верхній колонтитул* та *Нижній колонтитул* у вікні *Стиль сторінки* в *LibreOffice Writer*) використовують для налаштування колонтитулів.

ЦІКАВІ ДЖЕРЕЛА

Створення та оновлення змісту: *https://support.office.com/uk-ua/article/Створення-або-оновлення-змісту-1bee8114-2c58-46fb-a884-64c6dfecaeca*

Створення та оновлення покажчика: *https://support.office.com/uk-ua/article/Створення-й-оновлення-покажчика-cc502c71-a605-41fd-9a02-cda9d14bf073*

10. ПРАКТИЧНА РОБОТА 5

СТРУКТУРА ДОКУМЕНТА. АВТОМАТИЗОВАНЕ СТВОРЕННЯ ЗМІСТУ ТА ПОКАЖЧИКІВ

ПРИГАДАЙТЕ

- як за допомогою стилів визначити заголовки у великому документі;
- як переглянути структуру текстового документа;
- як додавати до текстового документа колонтитули, автоматизований зміст та покажчик;
- як форматувати параметри сторінки в текстовому документі.

СТВОРІТЬ

У власній структурі папок створіть папку *Практична робота 5.*

ПАМ'ЯТАЙТЕ

Під час виконання практичних завдань пам'ятайте про правила безпеки життєдіяльності при роботі з комп'ютером!

Завдання 1. Правопис основи слова (*10 балів*)

Визначте, які дії виконувалися при підготовці документа *Правопис основи слова*, що зберігається в папці *Текстовий процесор*:

- застосування стилів заголовків;
- додавання колонтитулів;
- розбиття документа на розділи;
- встановлення для окремих розділів різних параметрів сторінки;
- авоматизоване додавання змісту документа.

Відкрийте панель навігації та перегляньте структуру документа. Визначте, скільки рівнів заголовків виділено в документі. Скористайтеся заголовками на панелі *Навігація* для переміщення в те місце документа, де розпочинається кожний із розділів.

Перегляньте вміст документа. Визначте основні поняття, що використовуються в документі, та позначте їх як елементи покажчика. Наприкінці документа створіть покажчик.

Завдання 2. Авіаконтруктор Антонов (*10 балів*)

Внесіть зміни в документ *Антонов,* що зберігається в папці *Текстовий процесор*: застосуйте стилі до заголовків, створіть зміст, розбийте документ на розділи так, щоб зміст був розташований на окремій сторінці й кожний фрагмент тексту, що містить заголовок, розпочинався з нової сторінки. До розділу, що містить зміст, встановіть орієнтацію сторінки *альбомна* та значення лівого і правого полів — по 2 см. Додайте до документа колонтитул, що містить посилання на джерело матеріалу (посилання вказане наприкінці документа) та номери сторінок. Шаблон і розташування колонтитула (верхній чи нижній) оберіть самостійно.

Завдання 3. Твори Лесі Українки (*14 балів*)

За правилами створення складних документів створіть текстовий документ *Твори Лесі Українки,* який об'єднує збережені в папці *Текстовий процесор**Леся Українка* текстові документи, що містять поеми та вірші. Застосуйте до абзаців, що містять назви творів, стиль *Заголовок 2*. Додайте до документа колонтитули. Шаблон, вміст і розташування (верхній чи нижній) колонтитула оберіть самостійно. На початку документа створіть автоматичний зміст.

РОЗДІЛ 4

ОПРАЦЮВАННЯ ОБ'ЄКТІВ МУЛЬТИМЕДІА

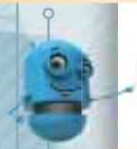
11. ОПРАЦЮВАННЯ АУДІО- ТА ВІДЕОФАЙЛІВ

ПРИГАДАЙТЕ:

- що розуміють під мультимедіа;
- які пристрої використовують для роботи з об'єктами мультимедіа;
- за допомогою яких програм можна переглядати відео та слухати аудіозаписи.

ВИ ДІЗНАЄТЕСЯ:

- які існують формати файлів звукозапису;
- які формати відеофайлів є найбільш поширеними;
- за допомогою яких програм можна перетворити звукозаписи та відеофайли з одного формату в інший;
- які існують сервіси для розміщення відео- та аудіоматеріалів в Інтернеті;
- як шукати та переглядати відео на *YouTube;*
- які програми використовують для опрацювання аудіозаписів;
- за допомогою яких програм можна створити власний відеокліп;
- які особливі елементи інтерфейсу має програма *Кіностудія.*



ВИВЧАЄМО

Мал. 11.1

1. Які існують формати файлів звукозапису?

Ви вже знаєте, що об'єкти мультимедіа (мал. 11.1) зберігаються у вигляді файлів, що містять мультимедійні дані. Формати аудіо- та відеофайлів визначають їхні структуру та способи кодування. Способи кодування звукових та відеоданих у різних форматах визначають якість звуку чи відтворення відео та ступінь стиснення даних, що впливає на обсяг файлів.

Існують такі найпоширеніші формати звукозапису (табл. 11.1):

Таблиця 11.1

Аудіоформат	Повна назва формату	Особливості формату
MIDI	***M**usical **I**nstrument **D**igital **I**nterface*	На відміну від інших аудіоформатів, являє собою не оцифрований звук, а набори вказівок (інструмент, ноти, що програються, значення параметрів звуку тощо), які можуть відтворюватися порізному, залежно від пристрою відтворення. Дає змогу обмінюватися даними між музичними інструментами, синтезаторами та комп'ютерами
WAV	***Wave**form audio format* (від англ. *wave* — хвиля)	Використовується в операційній системі *Windows*. Аудіоформат без використання стиснення. Точно передає звук, але займає значний обсяг на диску

Продовження таблиці 11.1

Аудіо-формат	Повна назва формату	Особливості формату
MP3	***MP**EG Layer **3***	Запис музики в цьому форматі відбувається зі стисненням обсягу з майже непомітним для слуху погіршенням якості, але обсяг зменшується в 10–12 разів порівняно з оригінальним музичним форматом. Принцип стиснення даних нагадує графічний формат *JPEG* — стиснення відбувається за рахунок виключення частот, які не сприймаються вухом людини
WMA	***W**indows **M**edia **A**udio*	Розроблений компанією *Microsoft* як альтернатива формату MP3. Ступінь стиснення даних і якість звуку майже аналогічні формату MP3. Нові версії формату, починаючи з *Windows Media Audio 9.1*, передбачають кодування без втрати якості, багатоканальне кодування об'ємного звуку та кодування голосу
AAC	***A**dvanced **A**udio **C**oding*	Під час кодування значно зменшується обсяг даних, потрібних для передавання високоякісного цифрового аудіо. У цьому форматі відбувається менша втрата якості, ніж у MP3, за однакових обсягів даних

2. Які формати відеофайлів є найбільш поширеними?

Формати відео є **медіа-контейнерами**, тобто можуть містити дані різних типів, стиснені різними кодеками, і дають змогу зберігати аудіо-, відео- і текстові дані (зокрема, субтитри) в одному файлі. Медіа-контейнер не лише надає можливість зберігати аудіо- й відеозаписи, а й забезпечує синхронізацію аудіо- та відеопотоків під час відтворення.

Відео може бути збережене на різних носіях. Як правило, відео високої якості має значний обсяг. Наприклад, музична комедія «Сорочинський ярмарок», записана на DVD, має обсяг 6,71 Гб.

Як і формати звукозапису, різні формати відеофайлів передбачають різні способи кодування даних, що визначають якість відео, ступінь стиснення даних та обсяг файла. Деякі формати можуть містити **потокове відео**, що використовується для передавання даних через Інтернет у режимі реального часу.

Існують такі поширені формати відео (табл. 11.2):

Таблиця 11.2

Відео-формат	Повна назва формату	Особливості формату
AVI	***A**udio-**V**ideo **I**nterleaved*	Може містити потоки чотирьох типів: відео, аудіо, MIDI, текст. Для стиснення аудіо- та відеозаписів можуть використовуватися різні кодеки. Має деякі обмеження, зокрема, обсяг файла не може перевищувати 4 Гб. На зміну цьому формату створено формат WMV

Цікаво

Кодек (від англ. *coder/decoder* — кодувальник/декодувальник) — пристрій чи програма, що виконує перетворення сигналів і використовується під час цифрового опрацювання відео та звуків для стиснення даних. Стиснення, як правило, відбувається із втратою якості. Кодеки дають змогу кодувати відеозаписи для передавання чи збереження, а також розкодовувати — для перегляду. Різні медіа-контейнери можуть підтримувати різні кодеки: DivX, XviD, MJPEG, VC-1 тощо.

Продовження таблиці 11.2

Відеоформат	Повна назва формату	Особливості формату
MPEG	***M**otion **P**icture **E**xperts **G**roup*	Розроблений *експертною групою з питань рухомого зображення* (*MPEG*). Було створено такі алгоритми стиснення даних: MPEG1, MPEG2 та MPEG4
MOV	*QuickTime **Mov**ie*	Один із перших відеоформатів, що набув широкого розповсюдження. Ступінь стиснення досить високий
ASF	***A**dvanced **S**ystems **F**ormat* (раніше також *Advanced Streaming Format, Active Streaming Format*)	Є частиною мультимедійного набору *Windows Media* для створення й розповсюдження аудіо- та відеофайлів. Формат файлів, що містять потокове аудіо та відео. Може використовуватись як для локального відтворення, так і для передавання та відтворення комп'ютерними мережами, зокрема Інтернетом. Особливістю формату є можливість відтворення безпосередньо в момент завантаження по мережі в режимі реального часу, що нагадує телевізійне мовлення, тобто *потокового відтворення*. Зазвичай використовується розширення файла *asf*, крім того, файли, що містять звукові записи, можуть мати розширення *wma*, а відеофайли — *wmv*
WMV	***W**indows **M**edia **V**ideo*	Є частиною мультимедійного набору *Windows Media*. Створений на основі формату AVI, але має додаткові можливості, зокрема, засоби захисту від несанкціонованого копіювання. Використовується для розповсюдження фільмів і відеокліпів
3GP	***3**rd **G**eneration* (*mobile*) ***P**hone*	Формат для збереження й перегляду відео на мобільних телефонах 3-го покоління. Відеозаписи в цьому форматі мають невеликий обсяг порівняно з іншими форматами відео, але за рахунок погіршення якості
RM	***R**eal **M**edia*	Розроблений компанією *RealNetworks* для розповсюдження відео через Інтернет. Ступінь стиснення даних та якість відео достатньо високі. Використовується для розповсюдження фільмів і трансляції так званого «інтернет-телебачення»
VOB	***V**ideo **Ob**ject*	Формат файлів, що використовується для збереження DVD-відео. Створений на основі MPEG2, може містити декілька потоків аудіо, відео, субтитри, а також меню фільма. Використовується для розповсюдження фільмів на DVD
FLV	***Flash Video***	Розроблений компанією *Adobe Systems*. Використовується для передавання відео через Інтернет, зокрема, такими сервісами: *YouTube*, *Вконтакте*, *RuTube* та ін. Файли в цьому форматі можна переглядати в більшості операційних систем, оскільки для цього використовується програвач *Adobe Flash Player*, який розповсюджується у вигляді *плагіна* для різних браузерів та різних операційних систем

Цікаво

Плагін (від англ. *plug-in* — підключати) — незалежний програмний модуль, що підключається до основної програми та призначений для розширення або використання її можливостей. Часто у вигляді плагіна виконується підтримка форматів файлів, наприклад, для звукових і відеопрогравачів, програм опрацювання звуку і графіки тощо. У веб-браузерах плагіни використовуються для забезпечення відображення форматів даних, які не мають вбудованої підтримки браузером (наприклад, *Adobe Flash* або *SVG*), для підлаштування можливостей до вимог користувача тощо.

3. За допомогою яких програм можна перетворити звукозаписи та відеофайли з одного формату в інший?

На практиці часто виникає потреба перетворити аудіо- чи відеофайли в інший формат. Наприклад, відеозаписи, зняті на мобільний телефон у форматі 3GP, для подальшого опрацювання можуть бути перетворені у формат AVI або WMV.

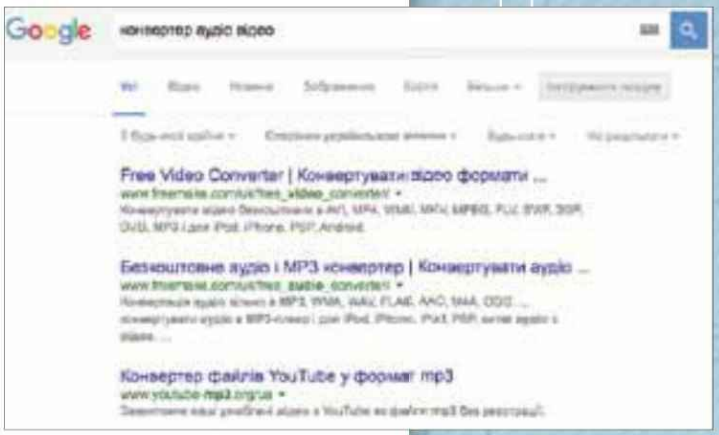
Для перетворення файлів з одного формату в інший використовують спеціальні програми — **конвертери**, вибір яких залежить від вихідного формату файла та формату, в який його потрібно перетворити.

Конвертер — програма, яка перетворює у файлі дані з одного формату в інший. Зміни і втрати даних, які можуть виникнути під час перетворення, залежать від форматів початкового та кінцевого файлів і від застосованої програми перетворення.

Існують різні програми, призначені для перетворення аудіо- чи відеофайлів, деякі з них є безкоштовними. Такі програми можна завантажити з Інтернету (мал. 11.2).

Після вибору одного з посилань на програми-конвертери, як правило, відображається опис програми та пропонується посилання для завантаження файла, за допомогою якого відбувається встановлення програми на комп'ютер.

Анотація дасть змогу визначити, які конвертери можна завантажити на вказаних сторінках, а також, чи є такі програми безкоштовними.



Мал. 11.2

4. Які існують сервіси для розміщення відео- та аудіоматеріалів в Інтернеті?

Сьогодні існують різні сервіси, на яких можна розміщувати, переглядати та прослуховувати відео- та аудіоматеріали. Деякі з них є безкоштовними, інші мають різні тарифні плани. Більшість сервісів для розміщення матеріалів передбачає попередню реєстрацію користувача.

YouTube — це найбільший у світі відеосервіс в Інтернеті, призначений для перегляду та розміщення в Інтернеті відео, створеного користувачами з усього світу. Переглядати відео на *YouTube* може будь-хто. На порталі *https://www.youtube.com* є система пошуку, за допомогою якої можна швидко знайти потрібні відеоматеріали.

Відео на сайті *YouTube* користувачі розміщують самостійно, також вони можуть залишати коментарі та оцінки. Правила спільноти й повідомлення на сайті свідчать про те, що користувачі мають володіти авторськими правами або мати дозвіл власників прав на розміщення будь-якого відеоматеріалу. Розміщувати відео, додавати коментарі та оцінки можуть лише зареєстровані користувачі.

Цікаво

Конвертер — від лат. *convertere* — перетворювати.

Цікаво

Компанія *YouTube* заснована в лютому 2005 р. Найперше відео на *YouTube* було розміщено 23 квітня 2005 р. У жовтні 2006 р. *Google* придбав *YouTube*, але за угодою за *YouTube* залишилися торгова марка й особливості бренду.

Список посилань на відео- та аудіосервіси

Існують також інші сервіси для розміщення та перегляду відео, зокрема *https://vimeo.com* та *http://rutube.ru*.

Деякі веб-портали мають вбудовані сервіси для розміщення аудіо- та відеоматеріалів. Наприклад, на порталі *http://meta.ua* є сервіси для розміщення відео й аудіо — *http://video.meta.ua* та *http://audio.meta.ua*, а також для пошуку та прослуховування радіо онлайн — *http://radio.meta.ua*.

Прикладами сервісів для розміщення та пошуку аудіозаписів є *https://podfm.ru/*, *http://www.audiomicro.com*, *http://freemusicarchive.org*, *http://boomp3.com*, *https://soundcloud.com*.

Аудіозаписи, змістом яких є не музика, а радіопередачі, звукові вистави, інтерв'ю, лекції тощо, можуть бути розміщені у вигляді подкастів. **Подкастом** називають або окремий аудіофайл, або серію таких файлів, що публікуються за однією адресою в Інтернеті та регулярно оновлюються.

Велика кількість сайтів надає можливість прослухати аудіозаписи онлайн. Це можуть бути улюблені пісні, аудіокниги, радіопередачі тощо. У цьому разі на веб-сторінці розміщений електронний плеєр з інструментами для початку чи призупинення прослуховування аудіозапису та регулювання гучності (мал. 11.3).

Цікаво

Термін **подкаст** є поєднанням назви портативного програвача музики *iPod* та слова *broadcast* — від англ. трансляція, радіомовлення.

Мал. 11.3

ДІЄМО

Вправа 1. Прослуховування аудіозаписів онлайн.
Завдання. Прослухайте фрагмент аудіозапису твору Лесі Українки «Давня казка».

1. Відкрийте вікно браузера й у рядку адреси введіть *https://lucaster.podfm.ru/ukraudio/140/*
2. Прослухайте за допомогою електронного плеєра (мал. 11.3) фрагмент аудіокниги «Давня казка» Лесі Українки у виконанні Ади Роговцевої.
3. Визначте, як змінити гучність відтворення, призупинити прослуховування.
4. Визначте тривалість аудіозапису.

5. Як шукати та переглядати відео на *YouTube*?

Щоб розпочати роботу з *YouTube*, слід завантажити браузер і в рядку адреси ввести *www.youtube.com*. Головна сторінка сайта, як і пошукових служб, містить рядок введення для ключових слів. У разі його використання пошук відбувається за вказаними ключовими словами серед відеозаписів, розміщених на *YouTube* (мал. 11.4).

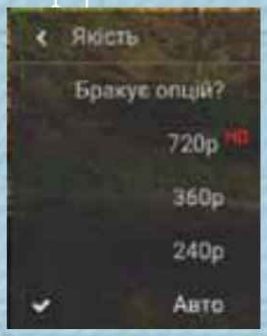
Мал. 11.4

Якщо ви знайшли відео, яке вас зацікавило, за його допомогою можна знайти й інші: у правій частині вікна є посилання на інші відео, розміщені цим користувачем, а також схожі відео.

У нижній частині області відтворення відеозапису (мал. 11.5) розміщено елементи керування відтворенням та переглядом: кнопки для початку або призупинення відтворення , регулювання гучності , переходу до режиму домашнього кінотеатру або повноекранного режиму . Перейти до перегляду наступного відео обраної тематики можна за допомогою елемента .

Мал. 11.5

Інструмент *Налаштування* обирають, щоб змінити якість відео залежно від швидкості інтернет-з'єднання (мал. 11.6).



Мал. 11.6

Відеозаписи, розміщені на *YouTube*, містять потокове відео, тому переглядати їх можна в режимі реального часу. Якщо швидкість завантаження даних з Інтернету недостатня для комфортного перегляду, можна призупинити перегляд за допомогою кнопки *Пауза* та дочекатися, доки буде завантажено хоча б третину чи половину відео.

ДІЄМО

Вправа 2. Пошук та перегляд відео на *YouTube*.
Завдання. Знайдіть і перегляньте на *YouTube* відео «Золота осінь. Аерозйомка».

1. Відкрийте вікно браузера й у рядку адреси введіть *www.youtube.com*.
2. На головній сторінці в рядок введення введіть ключові слова *аерозйомка осінь* або *аерозйомка Невицький замок*.
3. Оберіть один із знайдених результатів пошуку (*мал. 11.4*).
4. Перегляньте відеозапис. У разі потреби призупиніть відтворення та дочекайтеся часткового завантаження відео.
5. Серед схожих відео знайдіть інші відеозаписи про замки Карпат або аерозйомку. Перегляньте знайдені записи.

6. Які програми використовують для опрацювання аудіозаписів?

Створювати звукові файли та вносити зміни до аудіозаписів можна за допомогою **звукових редакторів**, або **аудіоредакторів**. Такі програми використовують, щоб записати звуковий файл із мікрофона, об'єднати кілька аудіозаписів, вирізати та зберегти фрагмент аудіозапису в окремому файлі, видалити паузи, позбавитися шуму тощо.

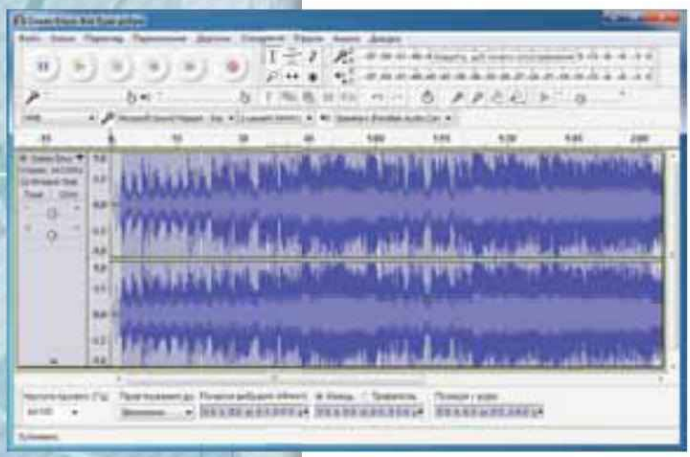
Існують як безкоштовні, так і більш потужні звукові редактори, які є платними. Прикладами звукових редакторів є *Audacity* (мал. 11.7), *mp3DirectCut*, *Ocenaudio*, *Adobe Audition*, *Acoustica*, *Audio Edit Deluxe* тощо.

Звукові дані в середовищі аудіоредактора графічно подаються у хвильовій формі. Область із таким графічним зображенням у звуковому редакторі називають **треком**, або **звуковою доріжкою**.

ДІЄМО

Вправа 3. Ознайомлення з призначенням та основними прийомами роботи з аудіоредактором *Audacity*.

Завдання. Визначте призначення деяких елементів управління аудіоредактора *Audacity*.



Мал. 11.7

Мал. 11.8

1. Виконайте вказівку *Пуск/Усі програми/Audacity*.
2. Розгляньте елементи вікна програми.
3. У меню *Файл* оберіть вказівку *Відкрити* та відкрийте файл *Океан Ельзи Все буде добре.mp3*, що збережений у папці *Об'єкти мультимедіа\Аудіо*.
4. Визначте призначення інструментів (мал. 11.8).
5. Прослухайте аудіозапис, скориставшись інструментом *Відтворення*. Визначте, коли розпочинається останній приспів.
6. Виконуючи протягування мишею, виділіть фрагмент аудіозапису, що містить останній приспів.
7. У нижній частині вікна перегляньте значення часу для властивостей *Початок вибраної області* та *Кінець*.
8. Ознайомтеся з вказівками, які містять різні пункти меню програми.

7. За допомогою яких програм можна створити власний відеокліп?

Для створення відеокліпів використовують відеоредактори.

Відеоредактор — це програма, що містить набір інструментів, за допомогою яких створюють і редагують відеофайли на комп'ютері.

Різні відеоредактори мають різні можливості й набори інструментів для опрацювання відеофайлів.

До складу операційної системи *Windows 7* входить відеоредактор *Кіностудія* (або *Windows Live Movie Maker*). Використовуючи його інструменти, можна видаляти зайві кадри з відеозапису, розташовувати відеофрагменти в будь-якій послідовності, додавати музичні файли, голосовий супровід, титри тощо. Відеоредактор *Кіностудія* було створено на зміну програмі *Windows Movie Maker*, що входить до складу операційної системи *Windows XP*.

Існують також інші відеоредактори, зокрема *VirtualDub, Pinnacle Studio, Free Video Editor, MS Producer, Adobe After Effects, Adobe Premiere, Ulead MediaStudio, SONY Vegas Pro* тощо. За допомогою програми *Camtasia Studio* можна також записувати відео з екрана комп'ютера.

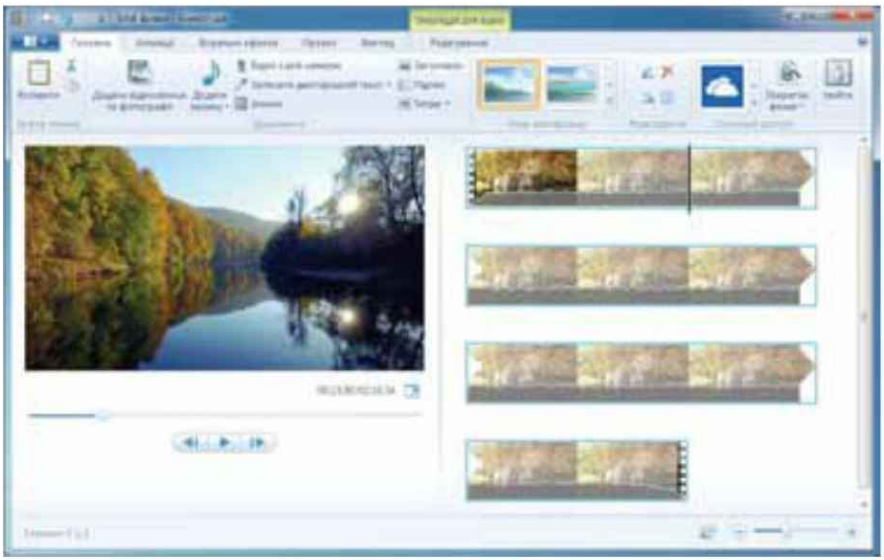
За допомогою відеоредактора створюється **проект** — файл, що містить відомості про порядок розташування та час відтворення аудіо- й відеокліпів, відеопереходи, відеоефекти, назви, титри тощо. Після збереження проекту його файл можна відкрити пізніше в середовищі відеоредактора

та внести до нього зміни. Проекти, створені у програмі *Кіностудія*, мають розширення *wlmp*, а у програмі *Windows Movie Maker* — *mswmm*.

Готовий проект може бути збережений як **фільм** — відеофайл у відповідному форматі, наприклад, WMV або MPEG4. Фільм можна зберегти на комп'ютері чи компакт-диску, відправити електронною поштою чи розмістити в Інтернеті.

8. Які особливі елементи інтерфейсу має програма *Кіностудія*?

Крім стандартних елементів вікна, зокрема, рядка заголовка та стрічки, що містить вкладки з інструментами, вікно відеоредактора *Кіностудія* містить також особливі елементи, притаманні відеоредакторам. Це **область попереднього перегляду**, що використовується для перегляду як окремих кліпів, так і всього проекту перед збереженням, а також область, у якій створюються й монтуються проекти, що використовується для перегляду та зміни послідовності кліпів проекту (мал. 11.9). Аудіокліпи, додані до проекту, відображаються нижче доданих відеокліпів.



Мал. 11.9

Цікаво

Якщо передбачається опрацювання відео, то комп'ютер має відповідати таким мінімальним системним вимогам: процесор потужністю не нижче 600 МГц, наприклад, *Intel Pentium III, AMD Athlon* тощо, 128 Мб ОЗП, 2 Гб вільного простору на диску, наявність пристрою звукозапису (для запису звуку із зовнішніх джерел) та пристрою відеозапису (для запису відео із зовнішніх джерел). Основною вимогою для можливості перенесення відео з відеокамери є наявність роз'єму *FireWire*, карти відеозахоплення або відеовходу на відеокарті.

ДІЄМО

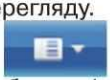
Вправа 4. Ознайомлення з призначенням та основними прийомами роботи з відеоредактором *Кіностудія*.

Завдання. Перегляньте відеозапис про призначення та основні прийоми роботи з відеоредактором *Кіностудія*.

1. Відкрийте папку *Об'єкти мультимедіа\Відео* та запустіть на відтворення файл *Огляд програми Кіностудія.wp4* (цей відеозапис можна також відкрити в Інтернеті на сторінці *https://www.youtube.com/watch?v=skyl8GFkabM*).
2. Перегляньте відеозапис.
3. Поясніть, які дії можна виконати за допомогою відеоредактора *Кіностудія*.

Вправа 5. Ознайомлення з елементами інтерфейсу відеоредактора *Кіностудія*.

Завдання. Визначте призначення деяких елементів управління відеоредактора *Кіностудія*.

1. Виконайте вказівку *Пуск/Усі програми/Кіностудія*.
2. Розгляньте елементи вікна програми.
3. За допомогою затримки миші визначте призначення інструментів на різних вкладках стрічки.
4. За допомогою затримки миші визначте призначення кнопок, розташованих у нижній частині області попереднього перегляду.
5. Відкрийте меню *Файл* за допомогою кнопки  в лівому верхньому куті вікна та ознайомтеся із вказівками роботи з файлами в програмі *Кіностудія*. Які команди є спільними й відмінними у програмі *Кіностудія* та текстовому процесорі *Microsoft Word*?

Обговоріть запитання, що містяться у файлі *Тема 11*, який зберігається у папці *Обговорюємо*. Також відкрити ці запитання можна за допомогою QR-коду.

ПРАЦЮЄМО В ПАРАХ

1. Користуючись Вікіпедією, знайдіть відомості про особливості:
   а) аудіоформатів;
   б) відеоформатів.
   Порівняйте відомості, наведені російською, українською та англійською мовами (за потреби скористайтеся перекладачем). Обговоріть у парах.
2. Обговоріть переваги та недоліки прослуховування аудіозаписів онлайн.
3. Знайдіть в Інтернеті веб-сторінки улюблених радіостанцій. Обговоріть, які з них надають можливість прослуховувати радіопередачі онлайн.
4. Чи можна імпортувати до проекту, що створюється за допомогою відеоредактора *Кіностудія*, відеозапис у форматі FLV? Обговоріть у парах.
5. Знайдіть та встановіть на комп'ютер одну з безкоштовних програм-конвертерів відео. Ознайомтеся з її інтерфейсом та можливостями. З'ясуйте, з яких форматів і в які можна виконувати перетворення за допомогою цієї програми. Навчіть одне одного користуватися програмою.

ПРАЦЮЄМО САМОСТІЙНО

1. Використовуючи словник або перекладач, перекладіть повні назви аудіо- та відеоформатів українською мовою. За потреби зверніться до вільної енциклопедії Вікіпедії.
2. Відкрийте веб-сторінку *http://www.elecard.com/mpeg/faq/index.php* (або файл *Експертна група MPEG* в папці *Об'єкти мультимедіа*). Ознайомтеся із запропонованими матеріалами та дайте відповіді на запитання:
   а) як часто збирається *експертна група з питань рухомого зображення* (MPEG);

б) у скільки разів відбувається стиснення відео при використанні формату MPEG;
в) як працює MPEG відео?

3. Ознайомтеся із особливостями створення та розміщення в Інтернеті подкасту. Скористайтеся посиланням *https://podfm.ru/topodcasters/*.
4. Знайдіть на *YouTube* відеозаписи, що відображають процес роботи чи результати діяльності людей різних професій: ландшафтного дизайнера, скульптора, телеведучого, журналіста, кулінара тощо. Збережіть посилання на знайдені сторінки до *Обраного*.

ДОСЛІДЖУЄМО

1. Дослідіть, як можна зберегти у файлі відео, знайдене на *YouTube*. За результатами дослідження підготуйте презентацію.
2. Визначте, файли яких типів можна використовувати під час створення відеокліпу за допомогою програми *Кіностудія*. Скористайтеся матеріалами за посиланням *http://windows.microsoft.com/uk-ua/windows-live/movie-maker-file-types-faq*.

ЦІКАВІ ДЖЕРЕЛА

Інструкції до роботи з програмою *Кіностудія*:
http://windows.microsoft.com/uk-ua/windows-live/windows-essentials-help#v1h=tab1
Путівник по звукових редакторах:
http://www.ixbt.com/soft/wave-editors-part1.shtml

12. СТВОРЕННЯ ТА НАЛАШТУВАННЯ ВІДЕО Й АУДІО

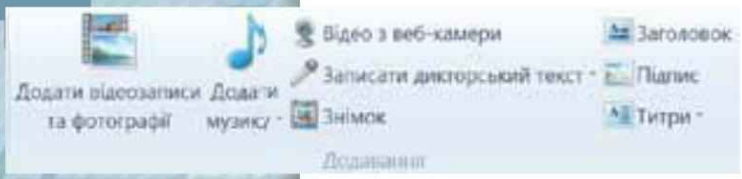
ВИ ДІЗНАЄТЕСЯ:

- як додати до проекту мультимедійні об'єкти та текстові написи;
- як розділити відеокліп на два та приховати початок чи кінець відеокліпу;
- як додати відеопереходи між кліпами чи зображеннями та візуальні ефекти;
- як додати до відеокліпу музичний супровід і налаштувати часові параметри відео та аудіо;
- як зберегти створений відеокліп у форматі відео;
- як опублікувати відео на *YouTube*;
- як можна вносити зміни до відео за допомогою онлайн-сервісів.

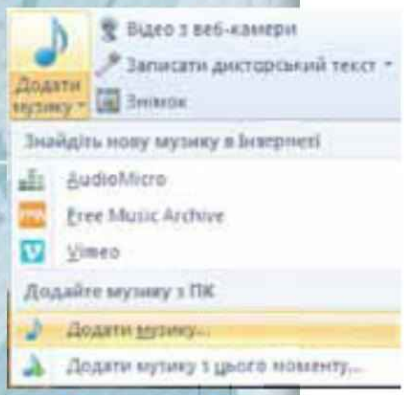
ВИВЧАЄМО

1. Як додати до проекту мультимедійні об'єкти та текстові написи?

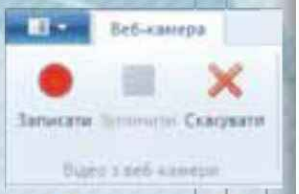
Для створення власного відео можна використати готові відеокліпи, збережені у файлах, або захопити відео безпосередньо з відеокамери чи веб-камери. Відеокліп, що створюється за допомогою програми *Кіностудія*, може містити відеофрагменти, аудіозаписи та статичні зображення. Для додавання до проекту мультимедійних об'єктів, що зберігаються у



Мал. 12.1



Мал. 12.2



Мал. 12.3

файлах, слід скористатись інструментами з групи *Додавання* вкладки *Головна* (мал. 12.1).

Якщо мультимедійні об'єкти збережені у форматах, що не підтримуються програмою *Кіностудія*, для використання в проекті їх слід спочатку конвертувати до одного з підтримуваних форматів.

Додавати музичний супровід до відеокліпу можна як з аудіофайлів, збережених на комп'ютері, так і з Інтернету (мал. 12.2).

Відеокліп можна доповнити звуковим супроводом у вигляді дикторського тексту, який може бути записаний із використанням мікрофону. Для запису й додавання до проекту відео чи дикторського тексту необхідно обрати відповідний інструмент у групі *Додавання* на вкладці *Головна*, та скористатись інструментом *Записати* для початку запису (мал. 12.3).

У процесі запису дикторського тексту в області попереднього перегляду відображається відеозапис, що дає змогу синхронізувати коментар із відео. Для завершення запису слід натиснути кнопку *Зупинити*, при цьому буде запропоновано ввести ім'я файла, до якого буде записано коментар. Створений відео чи аудіозапис зберігається у файлі та додається до проекту.

Сукупність доданих мультимедійних об'єктів стає вмістом проекту та майбутнім фільмом.

Використовуючи інструменти *Заголовок*, *Підпис*, *Титри* з групи *Додавання* вкладки *Головна* (мал. 12.1), можна додати до проекту текстові написи: назву фільму, пояснення чи коментарі до окремих кліпів, титри наприкінці фільму з відомостями про авторів, використані матеріали тощо.

ДІЄМО

Вправа 1. Створення відеокліпу, що складається з кількох відеозаписів.
Завдання. Створіть проект про музичні фонтани, що містить відеокліпи фонтанів в Одесі, Барселоні, Празі, Лас-Вегасі.

1. У власній структурі папок створіть папку *Мультимедіа*.
2. Відкрийте вікно відеоредактора *Кіностудія* та оберіть інструмент *Додати відеозаписи та фотографії*.
3. Додайте до проекту всі відеозаписи, що збережені в папці *Об'єкти мультимедіа\Відео\Музичні фонтани*.
4. Збережіть створений проект з іменем *Музичні фонтани* в папці *Мультимедіа* у власній структурі папок. Для цього натисніть кнопку та виконайте вказівку *Зберегти проект*.

Вправа 2. Додавання назви фільму та назв кліпів до проекту.
Завдання. До проекту *Музичні фонтани*, що збережений у папці *Мультимедіа* власної структури папок, додайте назву фільму та назви деяких кліпів.

1. Відкрийте проект *Музичні фонтани*, збережений у папці *Мультимедіа* власної структури папок.

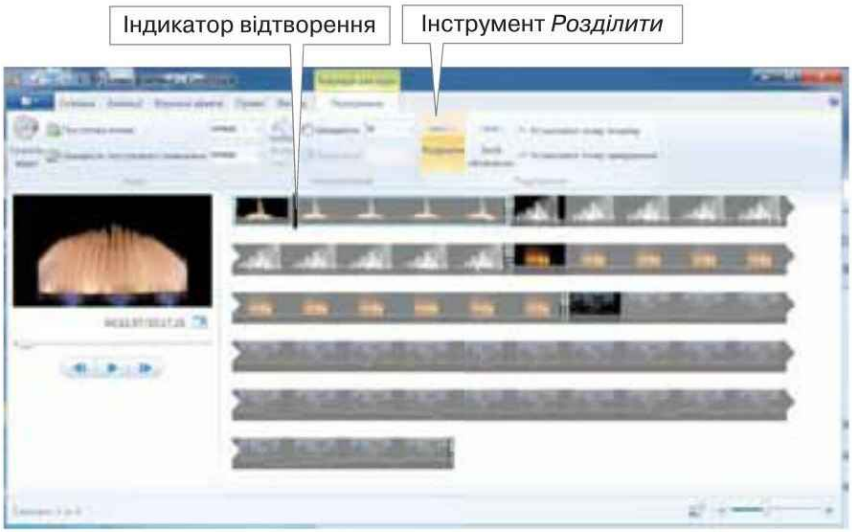
2. На вкладці *Головна* в групі *Додавання* оберіть інструмент *Заголовок*.
3. В області попереднього перегляду в текстовому написі введіть фразу *Музичні фонтани*. На вкладці *Форматування*, яка з'являється при опрацюванні текстових написів, змініть параметри форматування тексту на свій розсуд: задайте колір символів, їх розмір, накреслення тощо.
4. Виділіть відеокліп *Фонтан Одеса* й на вкладці *Головна* в області *Додавання* оберіть інструмент *Підпис*.
5. Введіть назву *Одеса* та встановіть параметри форматування для текстового напису.
6. Аналогічно додайте назви міст на початку кожного наступного відеозапису.
7. Перегляньте створений проект.
8. Виконайте вказівку *Зберегти проект*.

2. Як розділити відеокліп на два та приховати початок чи кінець відеокліпу?

Додані до проекту відеокліпи іноді потребують додаткового редагування — поділу на кілька частин та приховування початку чи кінця кліпу.

Якщо час відтворення кліпу достатньо великий і при цьому він вимагає редагування — вилучення кадрів чи додавання відеопереходу всередині кліпу, такий кліп можна розділити. Для поділу кліпу на два слід перемістити індикатор відтворення в місце кліпу, де його потрібно розділити, і на вкладці *Редагування* обрати інструмент *Розділити* (мал. 12.4).

Кліпи можуть містити фрагменти, які не потрібно відтворювати у фільмі. Якщо такі фрагменти розташовано на початку чи наприкінці кліпу, їх можна приховати — «обрізати» початок чи кінець кліпу. Для таких дій використовують інструмент *Засіб обтинання* на вкладці *Редагування*. Кожен виділений кліп має *точку початку* та *точку завершення* (мал. 12.5).



Мал. 12.4

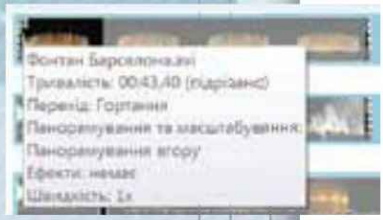
Мал. 12.5

Мал. 12.6

3. Як додати відеопереходи між кліпами чи зображеннями та візуальні ефекти?

Аналогічно до анімаційних ефектів зміни слайдів у комп'ютерних презентаціях, під час створення фільму можна додавати відеопереходи між окремими кліпами. Перелік відеопереходів міститься на вкладці *Анімації* (мал. 12.6). Для додавання відеопереходів до проекту достатньо виділити відеокліп та обрати один з ефектів переходу або панорамування й масштабування.

Іноді реалізувати задум автора допомагають візуальні ефекти, які можна додавати до кліпів. Наприклад, можна застосувати ефекти «старого фільму», ніби записаного на плівці, відтворити кліп у чорно-білому вигляді тощо. Перелік усіх доступних ефектів можна побачити на вкладці *Візуальні ефекти*. Перш ніж застосувати певний відеоефект до кліпу, можна переглянути, як він впливатиме на відтворення, для цього достатньо виділити відеоефект і переглянути його дію в області попереднього перегляду.

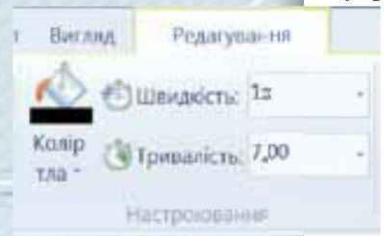


Мал. 12.7

Щоб додати до кліпу відеоефект, слід виділити відеокліп та обрати відповідний ефект. Дізнатися, на які з кліпів накладено відеоефекти чи додано відеопереходи, допоможе спливаюча підказка — якщо навести вказівник миші на кліп та виконати затримку, буде відображено список застосованих властивостей та їх значень (мал. 12.7).

4. Як додати до відеокліпу музичний супровід і налаштувати часові параметри відео та аудіо?

Відеозаписи, записані на відеокамеру чи імпортовані з файлів, можуть містити й звук. Якщо відеозапис не містить звуку або не потрібно відтворювати наявний звук, можна додати до відеокліпу інший музичний супровід чи звуковий коментар.



Мал. 12.8

Дуже рідко трапляється ситуація, коли тривалість відео та аудіо збігається, тому виникає необхідність налаштовувати часові параметри. Отримати відеозапис чи аудіокліп потрібної тривалості можна різними способами: приховати початок або кінець певних відео чи аудіокліпів, змінити швидкість відтворення відео в групі *Настроювання* на вкладці *Редагування* (мал. 12.8) відповідно для зменшення чи збільшення тривалості кліпу, додати до відеокліпу заголовок, титри або додаткові зображення тощо. Тривалість відображення зображень, заголовка та титрів можна змінювати за допомогою відповідної властивості (мал. 12.8) в групі *Настроювання*.

Мал. 12.9

Якщо тривалість аудіозапису довша, ніж відеоряду, аудіозапис буде автоматично «обрізано» наприкінці відеокліпа. Якщо аудіозапис відтворюватиметься не повністю, до нього можуть бути застосовані ефекти *Поступова поява* або *Швидкість поступового зникнення* в групі *Аудіо* на вкладці *Редагування* (мал. 12.9). Крім того, можна регулювати гучність відтворення звуку чи вимкнути звук для відеокліпу.

ДІЄМО

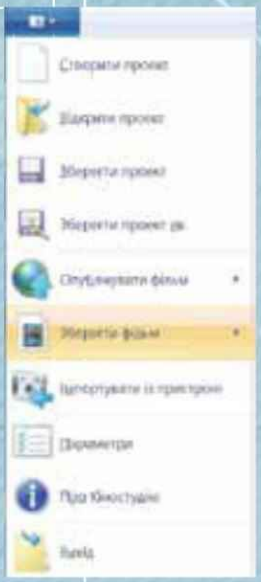
Вправа 3. Додавання до відеокліпу музичного супроводу та налаштування часових параметрів відео та аудіо.

Завдання. До проекту *Ранок*, що збережений у папці *Мультимедіа* власної структури папок, імпортуйте музичний файл *Гріг Ранок.mp3*. Додайте аудіокліп та налаштуйте часові параметри відео та аудіо.

1. Відкрийте вікно відеоредактора *Кіностудія* й оберіть інструмент *Додати відеозаписи та фотографії*.
2. Додайте до проекту відеозапис *Ранок.avi*, збережений у папці *Об'єкти мультимедіа\Відео*.
3. Оберіть інструмент *Додати музику* та імпортуйте звукозапис *Гріг Ранок.mp3*, що збережений у папці Об'єкти *мультимедіа\Аудіо*.
4. Щоб збільшити тривалість відеокліпу *Ранок* у 4 рази (уповільнити його), виділіть цей кліп і на вкладці *Редагування* в групі *Настроювання* встановіть *Швидкість* — 0,25.
5. Виділіть аудіокліп *Гріг Ранок* і на вкладці *Редагування* в групі *Аудіо* встановіть *Швидкість поступового зникнення* — висока (мал. 12.9).
6. Перегляньте створений відеокліп.
7. Збережіть створений проект у папці *Мультимедіа* власної структури папок.

5. Як зберегти створений відеокліп у форматі відео?

Після завершення роботи над проектом можна зберегти готовий відеокліп як фільм у форматі відео. Для цього слід виконати вказівку *Файл/Зберегти фільм* (мал. 12.10), виконання якої запускає *Майстер збереження фільмів*. Залежно від того, де потрібно зберегти фільм, може бути обрано один із запропонованих варіантів. На наступних кроках роботи *Майстра збереження фільмів* пропонується вказати ім'я файла та параметри, що впливають на якість відео й обсяг файла, але можна не змінювати параметри, запропоновані за замовчуванням.



Мал. 12.10

Збережений фільм може бути опублікований в Інтернеті (мал. 12.11).

Мал. 12.11

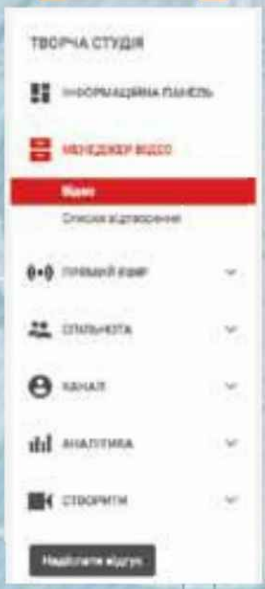
ДІЄМО

Вправа 4. Збереження створеного проекта відеокліпу у форматі відео.

Завдання. Збережіть проект *Музичні фонтани* як фільм.

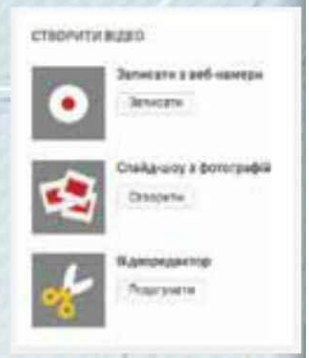
1. Відкрийте проект *Музичні фонтани*, збережений у папці *Мультимедіа* власної структури папок.
2. У меню *Файл* оберіть вказівку *Зберегти фільм* та оберіть параметри, рекомендовані для цього проекту.
3. У вікні збереження файла оберіть папку *Мультимедіа* власної структури папок, вкажіть назву фільму *Музичні фонтани*, оберіть розширення файла *wmv* та натисніть кнопку *Зберегти*.

Мал. 12.12



Мал. 12.13

Мал. 12.14



Мал. 12.15

6. Як опублікувати відео на *YouTube*?

Шукати й переглядати відео на *YouTube* можна навіть без реєстрації. Проте для завантаження відео необхідно скористатися власним обліковим записом *Google*. Щоб отримати можливість завантаження відео, слід увійти у свій обліковий запис та обрати вказівку *Завантажити* (мал. 12.12).

Перш ніж завантажувати відеозаписи, слід створити канал користувача. Це можна зробити у *Творчій студії* або під час першого використання вказівки *Завантажити*: буде запропоновано створити канал користувача, на якому зберігатимуться завантажені відеозаписи.

Відео, збережене у файлі, можна обрати в структурі папок або просто перетягнути мишею у вказану область.

Перш ніж опублікувати завантажене відео, необхідно вказати його назву, опис, ключові слова та інші налаштування для відео, за якими інші користувачі зможуть його знайти.

Переглянути список завантажених відео на каналі та внести до них зміни можна за допомогою *Менеджера відео* у *Творчій студії* (мал. 12.13).

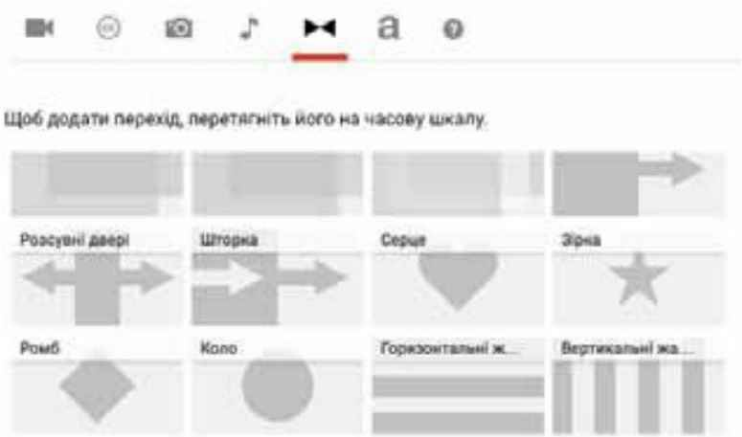
Кожне завантажене відео можна редагувати або видалити (мал. 12.14).

7. Як можна вносити зміни до відео за допомогою онлайн-сервісів?

Крім відеоредакторів у вигляді програм, що встановлюються на комп'ютер, існують також онлайнові відеоредактори, наприклад, *FileLab Video Editor* (*https://www.filelab.com/ru/video-editor*) та власний відеоредактор на *YouTube* — *https://www.youtube.com/editor*. Перейти до відеоредатора *YouTube* можна також зі сторінки для завантаження відео (мал. 12.15).

Як і в середовищі відеоредактора *Кіностудія*, в онлайновому відеоредакторі *YouTube* можна додати відеофрагменти, зображення, аудіозаписи, відеопереходи та текстові написи (мал. 12.16).

Відеофрагменти можна розділити на два, обрізати спочатку чи з кінця, виконати швидкі налаштування, повернути, якщо відеозапис було знято вертикально тощо.



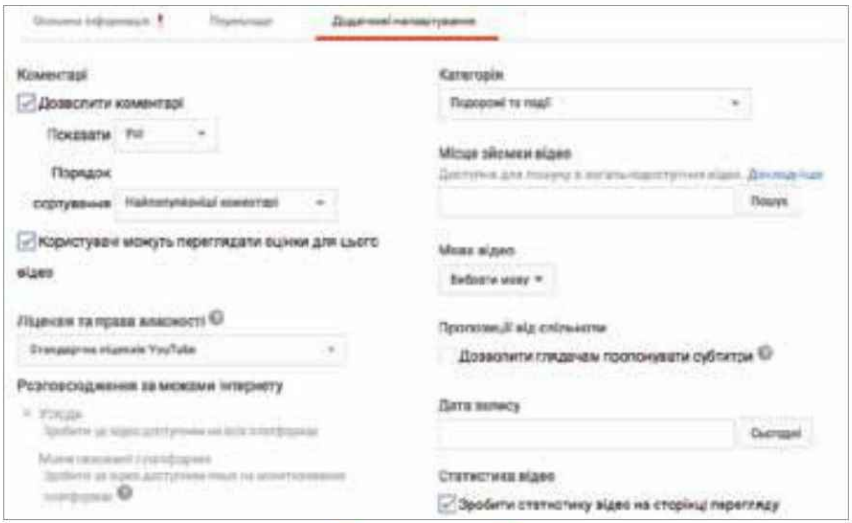
Мал. 12.16

ОБГОВОРЮЄМО

Обговоріть запитання, що містяться у файлі *Тема 12*, який зберігається у папці *Обговорюємо*. Також відкрити ці запитання можна за допомогою QR-коду.

ПРАЦЮЄМО В ПАРАХ

1. Розгляньте різні способи обтинання відеозапису: переміщення повзунків в області перегляду, задання часу точки початку й точки завершення, інструменти *Установити точку початку* та *Установити точку завершення* (мал. 12.5). Обговоріть, як використати кожний зі способів. Який спосіб є більш зручним для вас?
2. Що спільного й відмінного мають область попереднього перегляду відеоредактора *Кіностудія* та вікно програми *Медіапрогравач Windows*? За результатами порівняння побудуйте діаграму Венна. Обговоріть у парах.
3. Обговоріть переваги та недоліки редагування відеокліпу за допомогою онлайнового відеоредактора в порівнянні з відеоредактором, що встановлений на комп'ютері.
4. Ознайомтесь із додатковими налаштуваннями, які можна змінювати під час завантаження відео на *YouTube* (мал. 12.17). Обговоріть у парах, у яких випадках доцільно змінювати такі налаштування.



Мал. 12.17

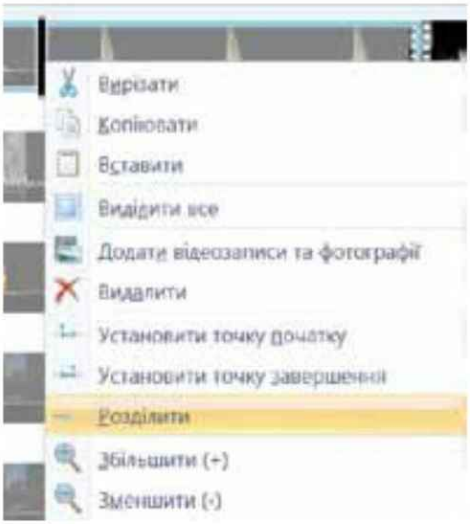
ПРАЦЮЄМО САМОСТІЙНО

1. Визначте, у яких папках за замовчуванням буде зберігатися відео, що захоплюється з відеокамери, та аудіозапис під час запису дикторського тексту.
2. Визначте, на яких сервісах можна розмістити в Інтернеті відео, створене в програмі *Кіностудія* (мал. 12.11).

3. Відкрийте проект *Музичні фонтани*, збережений у папці *Мультимедіа* власної структури папок. Додайте відеопереходи між кліпами на власний смак.
4. Створіть проект *Прогулянка Києвом*, до якого імпортуйте відеозапис *Прогулянка Києвом.wmv,* і застосуйте відеоефекти для відображення відео у відтінках сірого. Визначте, чи на весь кліп розповсюджується дія відеоефекту, доданого до кліпу, чи лише на його початок або певну частину.
5. Створіть відеоекскурсію своїм містом. Використайте фотографії, відеозаписи прогулянки, музичний супровід. Застосуйте відеопереходи й відеоефекти та створіть назву кліпу й титри. Збережіть фільм у файлі з іменем *Екскурсія.wmv* у папці *Мультимедіа* власної структури папок. Як приклад розгляньте відеозапис *Прогулянка Києвом.wmv*, що збережений у папці *Об'єкти мультимедіа\Відео*.
6. Створіть проект *Моє захоплення*, для якого підготуйте фотографії та відеозаписи. Використайте відеопереходи між зображеннями та відеозаписами, додайте назву фільму й титри. Готовий проект збережіть як файл фільму в папці *Мультимедіа* власної стурктури папок.

ДОСЛІДЖУЄМО

Дослідіть, які дії можна виконати з кліпом за допомогою вказівок контекстного меню (мал. 12.18).



Мал. 12.18

ЦІКАВІ ДЖЕРЕЛА

FileLab Video Editor:
https://www.filelab.com/ru/video-editor
Власний відеоредактор на *YouTube*:
https://www.youtube.com/editor

13. ПРАКТИЧНА РОБОТА 6

СТВОРЕННЯ ВІДЕОКЛІПУ. ДОДАВАННЯ ВІДЕОЕФЕКТІВ, НАЛАШТУВАННЯ ЧАСОВИХ ПАРАМЕТРІВ АУДІО- ТА ВІДЕОРЯДУ

ПРИГАДАЙТЕ

- як додавати у відеокліп аудіо- та відеофрагменти із зовнішніх джерел;
- як синхронізувати відеоряд з аудіорядом;
- як налаштовувати часові параметри аудіо- та відеоряду;
- як додавати відеоефекти до відеокліпу та налаштовувати переходи між його фрагментами.

СТВОРІТЬ

У власній структурі папок створіть папку *Практична робота 6.*

ПАМ'ЯТАЙТЕ

Під час виконання практичних завдань пам'ятайте про правила безпеки життєдіяльності при роботі з комп'ютером!

Завдання 1. Сім чудес Стародавнього світу (*9 балів*)

Знайдіть в Інтернеті відомості про Сім чудес Стародавнього світу. Використовуючи відеозаписи, збережені в папці *Об'єкти мультимедіа\Сім чудес світу*, створіть відеокліп про Сім чудес світу. Створіть назву фільму та додайте назви до кліпів про кожне із Семи чудес. Додайте відеопереходи між кліпами та відеоефекти.

Завдання 2. Ансамбль Вірського (*10 балів*)

Створіть новий проект у програмі *Кіностудія*. Додайте до проекту відеозаписи, збережені в папці *Об'єкти мультимедіа\Ансамбль Вірського*. Створіть відеокліп «Українські танці у виконанні ансамблю Вірського». Для цього з кожного танцю виберіть найбільш яскраві логічно завершені фрагменти (використайте поділ кліпу або приховування початку чи/та кінця кліпу). Додайте до кожного фрагмента назву танцю та відеопереходи між фрагментами.

Завдання 3. Ландшафтний дизайн (*12 балів*)

Перегляньте відеозаписи *Ландшафтний дизайн.avi* та *Ландшафтний дизайн Квіти.wmv*, що збережені в папці *Об'єкти мультимедіа\Ландшафтний дизайн*. Знайдіть в Інтернеті цікаві зображення, що стосуються ландшафтного дизайну. Доберіть музичну композицію у форматі, який підтримує відеоредактор *Кіностудія*, для звукового супроводу відеокліпу. Створіть проект про ландшафтний дизайн, до якого додайте знайдені зображення та окремі відеофрагменти із запропонованих відеозаписів. Застосуйте відеопереходи та відеоефекти. Додайте аудіозапис для звукового супроводу відеокліпу.

Завдання 4. Історія розвитку комп'ютерів (*12 балів*)

Використовуючи відеозаписи, збережені в папці *Об'єкти мультимедіа\Історія розвитку комп'ютерів*, створіть відеозапис про електронну обчислювальну машину ENIAC. Для цього створіть новий проект у програмі *Кіностудія*, додайте до проекту відеозапис *Історія розвитку комп'ютерів.mp4* і встановіть точку початку відтворення та точку завершення так, щоб відтворювався лише фрагмент про електронну обчислювальну машину ENIAC. Додайте до проекту відеозапис *Цікаві факти про перший комп'ютер.mp4*. Додайте дикторський текст, у якому будуть озвучені цікаві факти, відображені у відеозаписі.

14. ПРАКТИЧНА РОБОТА 7

РОЗМІЩЕННЯ АУДІО- ТА ВІДЕОМАТЕРІАЛІВ В ІНТЕРНЕТІ

ПРИГАДАЙТЕ

- як розміщувати аудіо та відеоматеріали в Інтернеті.

СТВОРІТЬ

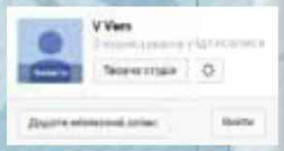
У власній структурі папок створіть папку *Практична робота 7*.

ПАМ'ЯТАЙТЕ

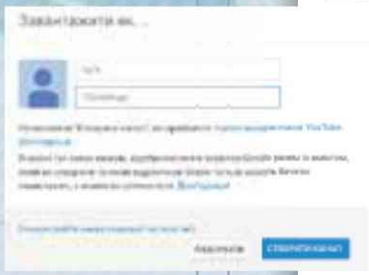
Під час виконання практичних завдань пам'ятайте про правила безпеки життєдіяльності при роботі з комп'ютером!

Завдання 1. Створення власного каналу на *YouTube* (*6 балів*)

1. Відкрийте вікно браузера й у рядку адреси введіть *https://www.youtube.com.* Увійдіть у власний обліковий запис *Google* за допомогою кнопки *Увійти* (мал. 12.14). Якщо у вас ще немає такого облікового запису, створіть його, скориставшись посиланням *Створити обліковий запис*. Оберіть об'єкт у правому верхньому куті вікна та натисніть кнопку *Творча студія* (мал. 14.1). Оберіть посилання *Створити канал.* У вікні, що з'явилося, введіть власне прізвище й ім'я, ознайомтеся з умовами використання *YouTube* та натисніть кнопку *Створити канал* (мал. 14.2).



Мал. 14.1



Мал. 14.2

Завдання 2. Публікація відео на *YouTube* (*11 балів*)

На сторінці сервіса *YouTube* переконайтеся, що ви увійшли у власний обліковий запис, та натисніть кноку *Завантажити* у правому верхньому куті вікна. Згорніть вікно браузера та відкрийте папку *Мультимедіа* у власній структурі папок. Розгорніть вікно браузера та зміни розміщення вікон так, щоб вікно браузера й вікно папки розміщувалися поруч на екрані. Перетягніть мишею файл *Музичні фонтани* в область для завантаження файлів (мал. 14.3). Введіть назву, опис відео та ключові слова для пошуку, за потреби змініть інші налаштування. Натисніть кнопку *Опублікувати.* Визначте URL-адресу сторінки, на якій опубліковане ваше відео. Надішліть електронною поштою адресу опублікованого відео вчителю.

Мал. 14.3

Завдання 3. Створення аудіозапису (*6 балів*)

За допомогою аудіоредактора запишіть із мікрофона уривок твору сучасного українського письменника чи поета. У разі потреби видаліть паузи з аудіозапису. Збережіть створений аудіозапис у файлі з іменем *Уривок твору.mp3* у папці *Практична робота 7.*

Завдання 4. Публікація аудіозапису (*4 бали*)

Зареєструйте власний обліковий запис на порталі *http://meta.ua.* Опублікуйте створений аудіозапис на сервісі розміщення аудіо *http://audio.meta.ua.*

Для цього скористайтесь інструментом *Завантажити*

. У вікні, що відкриється, в разі потреби змініть налаштування, вказані в кроках 1 та 2, і на кроці 3 оберіть файл *Уривок твору.mp3* з папки *Практична робота 7.* Дочекайтеся, поки файл буде повністю завантажений. Це можна визначити за допомогою відповідної шкали.

. Для публікації файла натисніть кнопку *Завантажити файли*

.

РОЗДІЛ 5

ОСНОВИ ПОДІЙНО- ТА ОБ'ЄКТНО-ОРІЄНТОВАНОГО ПРОГРАМУВАННЯ

15. МОВА ТА СЕРЕДОВИЩЕ ПРОГРАМУВАННЯ

ПРИГАДАЙТЕ:

- що називають алгоритмом та програмою;
- що або хто може бути виконавцем алгоритму;
- у якій формі можна подати алгоритм;
- що називають середовищем виконання алгоритму;
- як створюють програми та проекти в середовищі *Скретч*.

ВИ ДІЗНАЄТЕСЯ:

- що таке мова програмування;
- чим відрізняються мови програмування;
- які засоби необхідні для виконання створених програм;
- які середовища програмування використовують для створення програм;
- як працювати із проектом у середовищі програмування *PyCharm*.

ВИВЧАЄМО

1. Що таке мова програмування?

Ви вже знаєте, що алгоритми для виконавців описують різними способами та використовують різні форми їх подання. Якщо алгоритм створюють для виконавця людини, то, як правило, його подають словесно або графічно. Для опису алгоритмів, що орієнтовані для виконання за допомогою комп'ютера, використовують спеціальну **мову програмування**. Алгоритм, записаний мовою програмування, називають **програмою**.

Мова програмування — це система позначень для точного опису алгоритму, який потрібно виконати за допомогою комп'ютера.

Цікаво

Семантика — розділ мовознавства, що вивчає значення слів.

Мова програмування, як і будь-яка інша мова, має такі складові:

- набір символів, з яких утворюють слова та речення цієї мови — **алфавіт**;
- сукупність спеціальних слів, які мають однозначне пояснення та застосування — **словник**;
- систему правил складання базових конструкцій мови — **синтаксис**;
- правила **семантики**, що пояснюють, яке смислове значення має опис кожної з вказівок програми і які дії повинен виконати комп'ютер під час виконання кожної з таких вказівок.

Привіт!

Наприклад, щоб отримати повідомлення , яке ви створювали в навчальному середовищі *Скретч* за допомогою команди говорити Привіт!, мовою програмування *Python* потрібно ввести команду: `print('Привіт!')`. До алфавіту цієї мови належать латинські літери та спеціальні символи для запису команд, цифри — для числових даних, українські літери — для коментарів і текстових даних. Слово `print` належить до словника мови й задає команду *Друкувати*. Щоб надрукувати потрібний текст, його записують у дужках між символами « ' » — такі правила синтаксису.

Щоб команди програми могли виконуватися за допомогою комп'ютера, слід дотримуватись усіх правил обраної мови програмування. Якщо при складанні програми використати символи, які не входять до алфавіту обраної мови програмування, неправильно написати спеціальні слова чи скласти із цих слів структуру алгоритму, не дотримуючись прийнятих правил, це буде сприйнято як помилка і програма не виконається.

2. Чим відрізняються мови програмування?

Спочатку команди для комп'ютера писали за допомогою звичайних двійкових кодів і програма виглядала дуже громіздкою, її написання займало багато часу. З часом такі двійкові коди замінялись певними позначеннями, більш зрозумілими для людини. При цьому створювалися правила, а з часом і програми, які автоматично здійснювали переклад записаних таким чином програм для комп'ютера на мову машинних кодів. Сьогодні існує понад 3000 різноманітних мов програмування: деякі з них уже не використовуються, а інші, навпаки, ще й удосконалюються та дають поштовх розвитку нових мов і їх версій.

Мови програмування можна поділити на дві групи: мови **низького рівня** та мови **високого рівня** (мал. 15.1).

Машинна мова — набір команд, які виконуються безпосередньо центральним процесором (мал. 15.2). Працювати людині-програмісту з такою мовою важко через велику кількість команд у двійковій формі. Машинні коди були основним засобом програмування від моменту створення комп'ютерів.

До мов низького рівня належить **мова асемблера** (від англ. *assemble* — складати, компонувати). У мові асемблера використовуються символьні позначення команд, які легше зрозуміти й запам'ятати.

Програма, записана мовою програмування високого рівня, містить команди, що схожі на звичайні слова, записані англійською мовою. Наприклад, програма, записана мовою програмування *Python*, містить команди `print` — друкувати, `while` — поки, `if` — якщо тощо.

Програми, описані мовами високого рівня, дають можливість формулювати завдання для виконання на комп'ютері у звичному та зрозумілому для людини вигляді, а самі мови зробили використання комп'ютерів доступним широкому колу людей, які не є фахівцями в галузі програмування.

Розрізняють універсальні та спеціалізовані мови програмування високого рівня. Універсальні мови використовуються для вирішення різних завдань. До них відносять *C++*, *C#*, *Pascal*, *Delphi*, *Java*, *C*, *Objective-C*, *Basic*, *Swift*, *Python*, *Cobol*, *D*, *Ada*. Спеціалізовані мови призначені для вирішення завдань одного, максимум кількох видів завдань. Наприклад, для роботи з базами даних, веб-програмування або

Мал. 15.1

Машинний код		Асемблер	
0005	B4 09	7	mov AH, 09h
0007	BA0000r	8	mov DX, offset msg
00A	CD 21	9	int 21 h

Мал. 15.2

Мал. 15.3

написання скриптів для адміністрування операційних систем. Прикладами спеціалізованих мов програмування є *Visual FoxPro*, *PHP*, *Perl*, *VBScript*, *JavaScript*, *VBA* в *Microsoft Office* та інші.

За технологією програмування розрізняють (мал. 15.3):

- **процедурні** мови — програма складається із набору підзадач (процедур), які реалізують завдання;
- **об'єктно-орієнтовані** — головними елементами алгоритму є **клас** — новий тип даних, що розширює мову, й **об'єкт**, що має властивості та методи — дії, які виконуються об'єктом;
- **декларативні** — мови створення програм зі штучним інтелектом;
- мова **скриптів** (мова сценаріїв) — мова, розроблена для запису «сценаріїв» — послідовностей команд, які користувач може виконувати на комп'ютері.

Цікаво

Скриптова мова (англ. *scripting language*) — мова програмування, розроблена для запису «сценаріїв», послідовностей операцій, які користувач може виконувати на комп'ютері. У навчальному середовищі створення та виконання алгоритмів *Скретч* програми створюють в області, яка має назву *Скрипти*, бо вони власне реалізують сценарій подій, що мають відбутись із об'єктом-виконавцем.

ДІЄМО

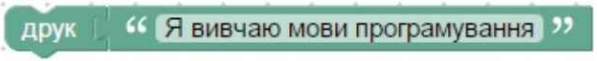
Вправа 1. Ознайомлення з мовами програмування.

Завдання. Реалізуйте в навчальному середовищі створення програм *Блоклі* (*https://blockly-demo.appspot.com/static/demos/index.html*) алгоритм, за яким у вікні повідомлення з'явиться текст *Я вивчаю мови програмування.*

1. Відкрийте вікно браузера. Введіть у поле адреси *https://blockly-demo.appspot.com/static/demos/index.html.*
2. На головній сторінці *Блоклі* оберіть середовище *Code Editor* (мал. 15.4).

Мал. 15.4

3. У списку обрання мови середовища English , розташованому у верхньому лівому куті веб-сторінки, оберіть українську мову.
4. Складіть програму із блоків групи *Текст* (мал. 15.5), так само, як ви робили це в середовищі *Скретч.*



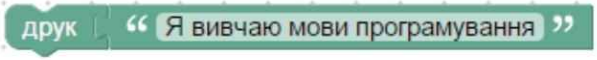
Мал. 15.5

5. Запустіть програму на виконання, натиснувши в лівому верхньому куті середовища кнопку *Запуск* . Перевірте, чи у вікні повідомлення відображено текст *Я вивчаю мови програмування*.

6. Перейдіть на сторінку кожної із запропонованих мов програмування (мал. 15.6).

Мал. 15.6

Ознайомтесь із текстом програми на кожній із мов програмування. Визначте, яка команда відповідає використаним блокам (мал. 15.6)



7. Закрийте вікно браузера.

3. Які засоби необхідні для виконання створених програм?

Для перетворення команд мови програмування, яка відрізняється від машинної мови, використовують спеціальні програми — **транслятори**.

Транслятор (від англ. *translation* — переклад) — програма, яка записує команди програми, описаної деякою мовою програмування, машинною мовою.

Розрізняють два способи трансляції: компіляція та інтерпретація.

Компілятор зчитує одразу всю програму й переписує її машинним кодом або мовою асемблеру. Процес трансляції, який при такому підході називається **компіляцією**, відбувається один раз, і результат перекладу зберігається в окремому файлі. Якщо код програми змінюється, її необхідно буде перекомпілювати. Скомпільована програма прив'язується до операційної системи та набору команд процесора, тому не завжди може бути перенесена й виконана на іншому комп'ютері. З іншого боку, вона є «готовою до використання» й може бути швидко виконана на тому самому або аналогічному комп'ютері: з точки зору користувача — просто клацнути на ім'я виконуваного файла й запустити на виконання, з «точки зору» комп'ютера — просто «прочитати» та виконати набір команд.

Інтерпретатор зчитує вихідний код програми по одній команді й одразу намагається їх «перекладати» та виконувати. Це дає змогу програмісту швидше перевіряти правильність виконання програми та знаходити помилки в коді. Виконання програми при застосуванні інтерпретатора займає трохи більше часу, оскільки кожного разу при запуску програми на виконання відбувається аналіз коду та його перетворення із самого початку програми.

Тому для підвищення швидкодії більшість сучасних інтерпретаторів насправді працює за змішаною схемою, спочатку транслюючи вихідний код програми в деяку проміжну форму — так званий байт-код. Це дає змогу, за відсутності змін в оригінальній програмі, не перечитувати її повністю, а використовувати байт-код як «напівфабрикат» для роботи. Виконання байт-коду все одно відбувається повільніше від машинного коду, але такий підхід є компромісом, який поєднує переваги інтерпретації та компіляції. Такий спосіб інтерпретації використовується в мові програмування *Python*, а також *Java* та *C#*.

Для створення програм, редагування, пошуку помилок та їх виправлення, а також виконання програм, написаних мовою програмування, використовують середовище програмування.

Цікаво

Розробка мови *Python* (вимовляють «пайтон») була розпочата наприкінці 1980-х років співробітником голландського інституту CWI Гвідо ван Россумом. Назва мови не пов'язана із відомим плазуном, хоча для неї й використовують позначку . *Python* має ефективні структури даних високого рівня і простий, але ефективний підхід до об'єктно-орієнтованого програмування. Елегантний синтаксис *Python*, динамічне опрацювання типів, а також те, що це інтерпретована мова, роблять її ідеальною для написання скриптів і швидкої розробки прикладних програм у багатьох галузях.

Середовище програмування — це комплекс програм, що містить засоби автоматизації процесів підготовки та виконання програм користувача, а саме:

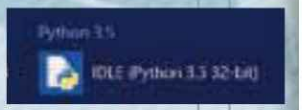
- редактор текстів програм — у ньому можна створювати й редагувати текст програми;
- довідково-інформаційну систему про мову програмування та середовище;
- бібліотеки, у яких зберігаються найбільш уживані фрагменти програм або цілі програми;
- компілятор чи інтерпретатор, використання якого дає змогу швидко знайти в програмі помилку та виправити її;
- покроковий «виконавець» програми.

4. Які середовища програмування використовують для створення програм?

Розрізняють такі види середовищ програмування: навчальне, інтегроване, візуальне (мал. 15.7).

Для однієї й тієї ж мови програмування може існувати декілька середовищ програмування, що підтримують різні технології програмування. Вони можуть бути встановлені на комп'ютер або реалізовані у вигляді онлайнового сервісу.

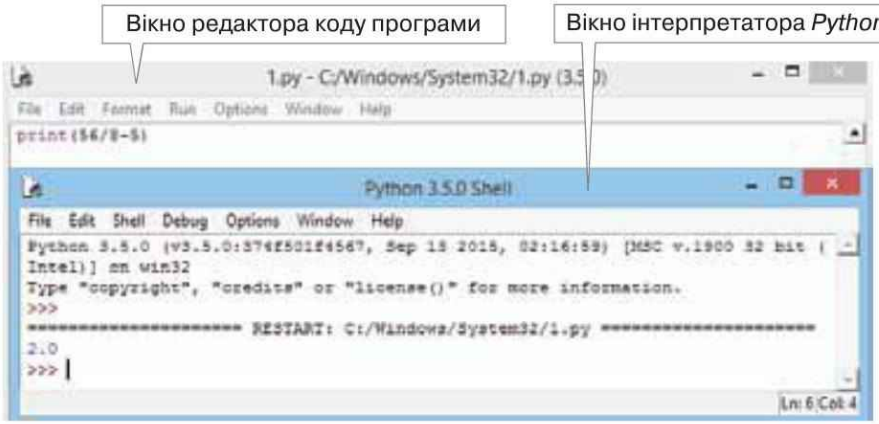
Середовища *Скретч* і *Блоклі* відносять до навчальних середовищ програмування.



При встановленні на комп'ютер інтерпретатора мови програмування *Python* одночасно також встановлюється інтегроване середовище програмування *IDLE*, яке можна відкрити з *Головного меню*. Особливістю цього середовища є те, що при роботі з ним використовуються два вікна, в одному з яких записується та редагується текст програми, а в іншому відображається результат її виконання. Після запуску програми *IDLE* за замовчуванням відкривається вікно інтерактивного середовища інтерпретатора *Python Shell*, у якому відображається результат виконання програми та повідомлення про помилки (мал. 15.8).

Якщо в цьому вікні в меню *File* (*Файл*) обрати вказівку *New File* (*Новий файл*), то відкриється вікно редактора коду програми, у якому можна створювати, зберігати, переглядати, редагувати, запускати програму, написану мовою програмування *Python*. Наприклад, щоб відобразити результат обчислення значення виразу *56 : 8 – 5*, достатньо у вікні редактора коду програми використати команду `print ()`, де в дужках записати вираз із використанням знаків арифметичних операцій: + (додавання), – (віднімання), * (множення) та / (ділення). Результат виконання такого

Мал. 15.7



Мал. 15.8

фрагмента програми після його збереження та запуску за допомогою вказівки *Run* буде відображено у вікні *Python Shell* (мал. 15.8).

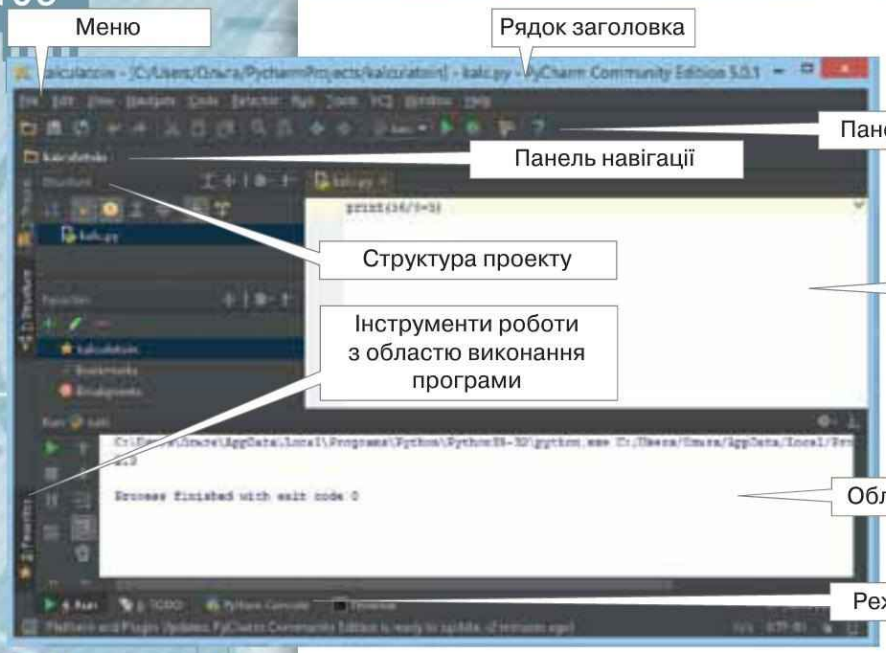
Для створення та виконання програм, написаних мовою програмування *Python*, можна використовувати онлайнове середовище програмування *CodingGraund*. Після аналізу системи на екрані з'являється номер рядка, у якому можна вводити команди мовою *Python*. Наприклад, на малюнку 15.9 наведено фрагмент програми для обчислення виразу, розглянутий у попередньому прикладі.

(*http://www.tutorialspoint.com/ipython_terminal_online.php*)

Зручним для використання інтегрованим середовищем програмування, яке підтримує сучасні складові мови *Python*, є *PyCharm*. Це середовище поєднує редактор для введення й редагування програми, транслятор і налагоджувач помилок. Користувач отримує можливість використовувати статичний аналіз коду, підсвічування синтаксису та помилок, навігацію серед проектів і коду програми, відображення файлової структури проекту, швидкий перехід між файлами, класами, методами та багато іншого.

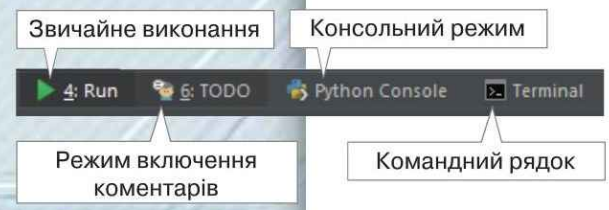
Вікно середовища складається з рядка заголовка, меню, панелі інструментів, панелі навігації, структури проекту, редактора коду програми,

Мал. 15.9



Мал. 15.10

Мал. 15.11



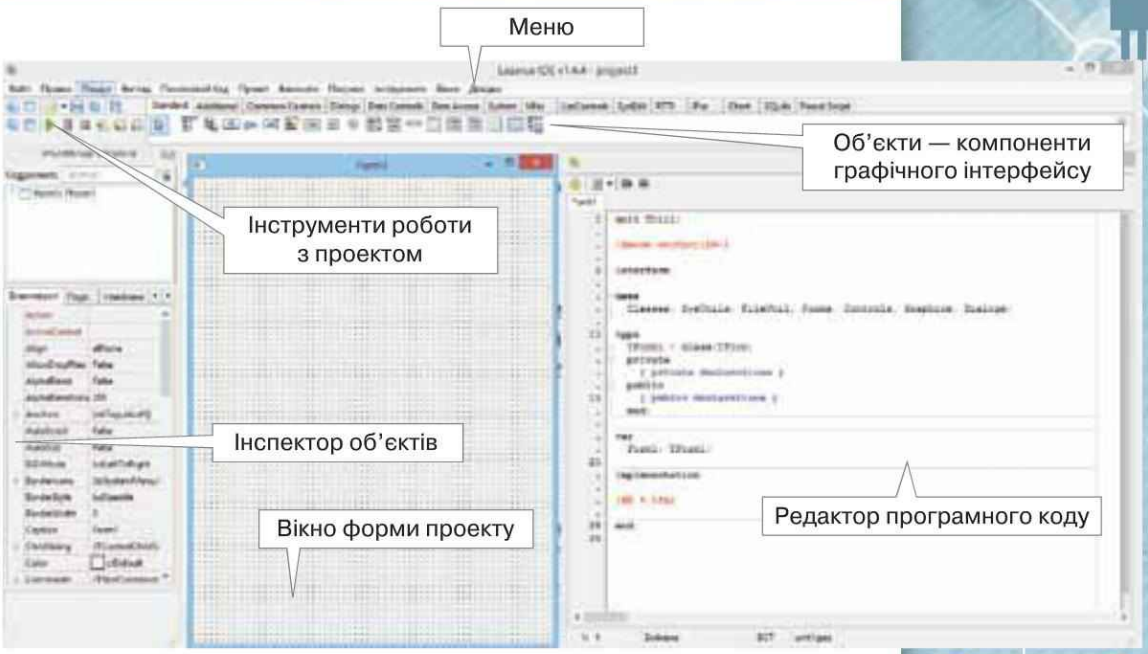
Мал. 15.12

області виконання програми, інструментів роботи з областю виконання програми, режимів роботи середовища (мал. 15.10).

Інтегроване середовище програмування *PyCharm* містить набір засобів допомоги та підтримки користувача. Після завантаження середовища з'являється інформаційне вікно, у якому можна дізнатися про корисні послуги при роботі з програмою та комбінації клавіш, якими вони викликаються. Крім цього, при введенні команди в редакторі коду програми одразу після введення з клавіатури першої літери команди стає доступним список службових команд із підказками про їх синтаксис (мал. 15.11).

Область виконання програми підтримує декілька режимів, які можна змінювати за допомогою відповідних інструментів: звичайне виконання, режим включення коментарів, консольний режим, який нагадує середовище *IDLE*, командний рядок (мал. 15.12).

Для спрощення розробки графічного інтерфейсу користувача використовують середовища для візуального програмування, які дають змогу конструювати програми шляхом оперування графічними об'єктами. Для мови *Python* розроблені спеціальні бібліотеки, за допомогою яких можна візуалізувати процес розробки проекту. Візуальні компоненти графічного інтерфейсу проекту можна також створювати вручну, використовуючи в програмі відповідні команди.



Мал. 15.13

Прикладом візуального середовища програмування є *Lazarus*, що підтримує мову програмування *Free Pascal* (мал. 15.13). Для візуалізації інтерфейсу проекту, створеного в цьому середовищі, існує цілий ряд спеціально розроблених елементів інтерфейсу — візуальних компонентів, що дають змогу відображати різні дані — здійснювати управління програмою загалом. Найпростіший приклад — кнопка у вікні програми. Об'єкт *кнопка* імітує поведінку звичайної кнопки на пульті управління будь-якого приладу. Її можна «натискати» за допомогою миші. Вікно форми проекту, у якому проектується інтерфейс, та вікно, у якому створюється програма, — розділені.

Таке середовище нагадує відоме вам середовище *Скретч*, у якому в області скриптів складають програму, а на сцені розташовують об'єкти, з якими після запуску проекту на виконання відбуваються події, що відповідають командам програми.

5. Як працювати із проектом у середовищі програмування *PyCharm?*

У середовищі програмування *PyCharm* можна створювати декілька різних об'єктів:

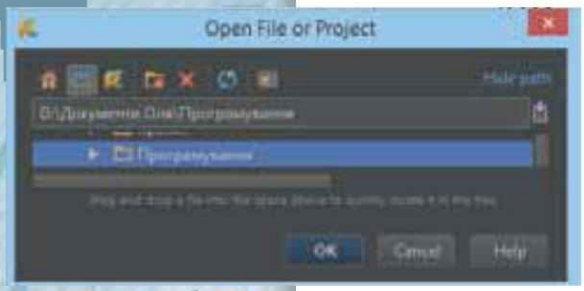
- файл, який можна використовувати, наприклад, для збереження даних

;
- папку, де можна зберігати, наприклад, декілька проектів чи допоміжних програм, які потім можна використовувати як файли-заготовки для розробки нових проектів

;
- пакети

;
- файл програми на мові *Python*

;



Мал. 15.14

- скрипт або деяку сторінку, яку можна розмістити в Інтернеті HTML File.

Усі ці об'єкти та їх групи можна об'єднати в одному проекті, який створюють за допомогою вказівки меню *New Project* (*Новий проект*) із меню *File* (*Файл*).

Файл програми на мові *Python* має розширення *py*. Щоб його відкрити в середовищі *PyCharm*, використовують вказівку *Open* (*Відкрити*) із меню *File* (*Файл*). У вікні відкривання файла чи проекту вказують шлях до шуканого файла та натискують кнопку *OK* (мал. 15.14).

Щоб виконати програми, записані у файлах проекту, використовують вказівку *Run* (*Виконати*) з панелі інструментів або з однойменного меню.

ДІЄМО

Вправа 2. Створення проекту в середовищі програмування *PyCharm*.
Завдання. Створіть проект *Діалог* у середовищі *PyCharm*. Складіть програму введення й виведення власного імені та прізвища на мові *Python*.

1. Завантажте середовище *PyCharm* одним із відомих способів запуску програми.
2. Ознайомтесь із запропонованою довідкою та закрийте вікно *Tip of the day* (*Порада дня*).
3. У меню *File* (*Файл*) оберіть вказівку *New Project* (*Новий проект*).
4. У діалоговому вікні *Create Project* (*Створення проекту*) оберіть папку для розміщення проекту й замість імені за замовчуванням *untitled1* задайте власне ім'я — *dialog* (мал. 15.15). Натисніть кнопку *Create* (*Створити*).

Мал. 15.15

Мал. 15.16

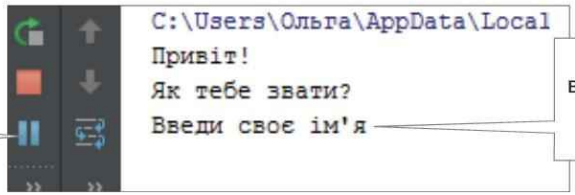
5. Створіть новий файл у проекті *dialog*. Для цього в меню *File* (*Файл*) оберіть вказівку *New* (*Новий*), а в переліку об'єктів — Python File. У вікні *Новий файл* задайте ім'я файлу *Привіт*. Натисніть кнопку *OK* (мал. 15.16).
6. Введіть текст програми у вікні редактора коду (мал. 15.17). Зверніть увагу: якщо текстове повідомлення українською мовою містить символ апостроф ('), то перед ним слід записати символ «\». У цьому разі апостроф буде сприйматися як частина тексту, а не вказувати на завершення текстового фрагмента.

Рядок, що починається символом # — це **коментар** у тексті програми. Коментарі використовують, щоб допомогти тому, хто ознайомлюється з текстом програми. Вони не впливають на хід її виконання

```
# Запиши текст програми за зразком
print('Привіт!')
print('Як тебе звати?')
name = input('Введи своє ім\'я')
print('Радий тебе вітати,', name)
```

Мал. 15.17

7. Запустіть проект на виконання. Для цього оберіть інструмент *Run* (*Виконати*) .
8. Зверніть увагу на нижню частину вікна середовища та введіть дані, які потрібні для виконання програми (мал. 15.18).



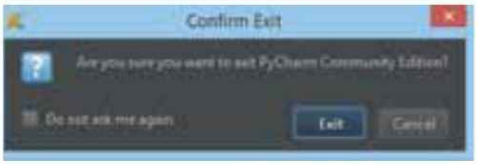
Мал. 15.18

9. Перевірте, чи завершилось виконання всіх команд — на екрані має з'явитися повідомлення `Process finished with exit code 0`.
10. Очистіть область виконання програми. Для цього оберіть інструмент — *Clear All* (*Очистити все*).
11. Додайте до програми рядки так, щоб після запуску програми отримати повідомлення, які, наприклад, отримав Василь Антоненко (мал. 15.19).
12. Збережіть зміни у проекті. Для цього застосуйте інструмент — *Save* (*Зберегти*) на панелі інструментів.
13. Закрийте вікно середовища. Для цього у меню *File* (*Файл*)оберіть вказівку *Exit* (*Вийти*). У вікні підтвердження виходу натисніть кнопку *Exit* (*Вийти*) (мал. 15.20).
14. Підтвердьте завершення роботи з програмою *Привіт* — оберіть кнопку Disconnect.

```
Привіт!
Як тебе звати?
Введи своє ім'я Василь
Радий тебе вітати,  Василь
Яке твоє прізвище?
Введи своє прізвище Антоненко
Будьмо знайомі,  Василь  Антоненко

Process finished with exit code 0
```

Мал. 15.19



Мал. 15.20

Тут відсутній символ « ' »

```
print('Привіт!)
print('Як тебе звати?')
name = input('Введи
своє ім'я')
```

Тут відсутній символ «\» після «ім»

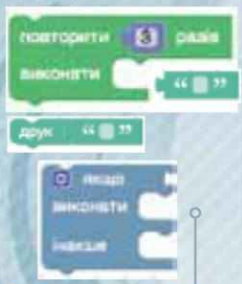
Мал. 15.21

ДОСЛІДЖУЄМО

Вправа 3. Помилки в програмі мовою *Python*.
Завдання. Відкрийте проект *Dialog* у середовищі програмування *PyCharm*. Змініть текст програми у файлі *Привіт*, як показано на малюнку 15.21. Перевірте, як у середовищі програмування повідомляють користувача про помилку та як її виправлення впливає на результат. Зробіть висновки.

ОБГОВОРЮЄМО

Обговоріть запитання, що містяться у файлі *Тема 15*, який зберігається у папці *Обговорюємо*. Також відкрити ці запитання можна за допомогою QR-коду.



Мал. 15.22

ПРАЦЮЄМО В ПАРАХ

1. У середовищі *Блоклі* створіть програму, яка у вікні повідомлення тричі виводитиме на екран деякий текст. Обговоріть, які із запропонованих блоків (мал. 15.22) можна для цього використати.
   Яким командам у середовищі *Скретч* відповідають обрані блоки? Визначте, як команди повторення та друку реалізовуються в різних мовах програмування.
2. Знайдіть в Інтернеті відомості про історію виникнення різних мов програмування. Обговоріть і сплануйте, як можна було б подати знайдені відомості на лінії часу та в якій програмі зручно було би створити таку лінію часу. Створіть її.
3. Обговоріть, що спільного та які відмінності є в навчальних середовищах програмування *Скретч* і *Блоклі*. Результат обговорення подайте діаграмою *Венна*.
4. Обговоріть, у яких випадках ви б порадили своїм однокласникам для вивчення мови *Python* використовувати середовище програмування *IDLE*, онлайнове середовище *CodingGraund*, інтегроване середовище програмування *PyCharm*. Оберіть хоча б три аргументи для підтвердження своїх порад.

ПРАЦЮЄМО САМОСТІЙНО

1. Визначте, що спільного та відмінного має навчальне середовище програмування *Скретч* і середовище візуального програмування *Lazarus*. Використайте відомі засоби текстового процесора, щоб наочно подати свою відповідь.
2. Складіть мовою програмування *Python* програму, за допомогою якої можна отримати результати знаходження значень виразів:
   1) $125 : 25 + 10$;
   2) $2 \cdot 3 \cdot 4 \cdot 5 : 6$;
   3) $765 : 9 + 48 \cdot 4 - 121$.
   Запишіть ці вирази за правилами математики. Знайдіть значення записаних виразів. Порівняйте отримані результати «ручного» обчислення та за допомогою складеної програми. Зробіть висновки про правила запису дій мовою програмування *Python*.
3. Відкрийте в середовищі програмування *PyCharm* програму *Guess.py,* що збережена в папці *Програмування.* Запустіть її на виконання. Введіть дані, що відповідають запитам програми. Визначте, яке завдання реалізоване в цій програмі.
4. Складіть програму *Термінал* мовою програмування *Python*, у якій отримують повідомлення про поповнення рахунку оператора мобільного зв'язку. Наприклад:
   Введіть назву оператора мобільного зв'язку Київстар
   Введіть номер телефону для поповнення рахунку 0970707007
   Вкажіть суму поповнення рахунку 20
   Шановний абоненте Київстар! Ви поповнюєте телефон за номером 0970707007 на суму 20 грн.

16. ОБ'ЄКТИ ПРОГРАМ ІЗ ГРАФІЧНИМ ІНТЕРФЕЙСОМ

ПРИГАДАЙТЕ:

- що таке об'єкт та чим відрізняються об'єкти;
- які особливості має графічний інтерфейс програми;
- які об'єкти графічного інтерфейсу використовують в операційній системі *Windows*;
- як створюють проекти в середовищі *Скретч*.

ВИ ДІЗНАЄТЕСЯ:

- які особливості мають програми з графічним інтерфейсом;
- які є способи створення об'єктів графічного інтерфейсу;
- як працювати з проектами у середовищі програмування *Lazarus*;
- як працювати з екранною формою в середовищі *Lazarus*;
- як у середовищі *Lazarus* розробити прикладну програму.

ВИВЧАЄМО

1. Які особливості мають програми з графічним інтерфейсом?

Вам уже доводилося працювати з різними програмами, що мають графічний інтерфейс: редакторами, тренажерами, програмами для виконання обчислень, іграми тощо. Вони розроблені за допомогою певних середовищ програмування та мають спільні ознаки:

- програма відкривається у вікні, розмір якого, як правило, можна змінювати;
- графічний інтерфейс програми, який містить зображення значків, меню, кнопки, текстові поля тощо, дає змогу користувачеві за допомогою миші виконувати потрібні команди, змінювати значення властивостей об'єктів і вводити за допомогою клавіатури текстові та числові дані;
- усі об'єкти, використані у програмі, мають властивості, які можна змінювати;
- події, що відбуваються у програмі, пов'язані з певними об'єктами. Наприклад, з об'єктом *меню програми* пов'язана подія, яка викликається натисненням лівої кнопки миші на обраному розділі — розгорнути список вказівок, що об'єднані в цьому розділі меню.

В основу розробки програм, які ви використовували для розв'язування різних прикладних завдань, покладена парадигма **подійно-орієнтованого програмування**: програма — це сукупність об'єктів реального або віртуального світу, що мають певні властивості та з кожним з яких пов'язаний деякий набір подій (мал. 16.1).

Виконання кожної події приводить до деяких очевидних чи прихованих змін певного об'єкта відповідно до написаного програмного коду, який створюється користувачем для кожного об'єкта окремо. Програмний код пов'язаний із вікнами та елементами управління програмного середовища. Вам знайомі різні елементи управління вікна програми, такі як кнопки, прапорці, перемикачі, списки, поля введення, смуги прокручування тощо. Більшість із них призначено для введення даних користу-

Цікаво

Парадигма програмування — це система ідей і понять, які визначають стиль написання програмного коду, а також спосіб мислення програміста.

Мал. 16.1

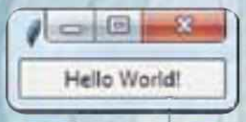
вачем та виведення результатів роботи програми на екран (або на папір, або у файл), а також для управління роботою комп'ютера.

У програмному коді, що орієнтований на опрацювання подій, програміст повинен вказати, як слід реагувати на різні події (чи дії користувача). Це можуть бути, наприклад, такі події: вибір вказівки, клацання кнопкою миші, переміщення миші тощо.

2. Які є способи створення об'єктів графічного інтерфейсу?

Створювати об'єкти графічного інтерфейсу в середовищах програмування можна двома способами:

1) безпосередньо в редакторі коду програми;
2) використовуючи засоби графічного інтерфейсу середовища візуального програмування.



Мал. 16.2, *а*

```
# підключення модуля віконного
графічного інтерфейсу користувача
import tkinter
# створення вікна програми
main = tkinter.Tk()
# створення текстового напису
label = tkinter.Label(text="Hello World!")
# розміщення напису на головній формі
label.pack()
# запуск опрацювання подій програми
main.mainloop()
```

Мал. 16.2, *б*

Перший спосіб передбачає підключення додаткових програм — **модулів**, що містять команди для створення графічного інтерфейсу, які можна використовувати в програмному коді. Створення об'єктів програми, їх розташування та запуск опрацювання подій програми задаються командами мови програмування. Наприклад, найпростіша програма із графічним інтерфейсом користувача — виведення повідомлення *Hello World!* у вікні (мал. 16.2, *а*), яка створена в середовищі програмування *PyCharm*, складається з команд, зображених на малюнку 16.2, *б*.

У середовищі візуального програмування кожний програмний проект складається з інтерфейсу користувача, який забезпечує введення та виведення даних, управління їх опрацюванням, а також із програмного коду — програми (мал. 16.3). Результатом виконання програми з графічним інтерфейсом є екранна форма, що містить об'єкти. Програма може опрацьовувати одну або декілька екранних форм.

Мал. 16.3

Інтерфейсом користувача у програмі називається візуалізована частина програми, за допомогою якої подаються відомості користувачеві та приймаються від нього відомості для управління роботою програми.

Існують різні інтерфейси користувача. В ОС *Windows* прийнято стандартний графічний інтерфейс, у якому користувач використовує мишу. Так, наприклад, інтерфейс користувача програм для ОС *Windows* може складатися з меню, однієї чи кількох панелей інструментів чи стрічок (наборів кнопок з малюнками) для прискорення виконання дій, і робочої області, що має вигляд документа або розгорнутого аркуша. До інтерфейсу програм, що працюють під управлінням ОС *Windows*, належать такі основні об'єкти, як **вікна** (прикладних програм, повідомлень, документів і діалогові), **панелі інструментів** та **меню**, кожний з яких має також свої елементи. Їх сукупність утворює **візуальну складову** будь-якої прикладної програми. Формувати таку візуальну складову можна за допомогою простих спеціальних засобів графічного редагування (компоновки), наприклад, у середовищі програмування *Lazarus*.

ДІЄМО

Вправа 1. Вікно повідомлення.

Завдання. Створіть програму в середовищі програмування *PyCharm*, за допомогою якої у вікні повідомлення буде виводитися цитата академіка В. М. Глушкова:

```
Людина, яка у XXI столітті не вмітиме користуватися комп'ютером,
буде подібна до людини XX століття, що не вміє ні писати, ні
читати.
```

1. Запустіть середовище програмування *PyCharm*.
2. Створіть новий файл програми мовою *Python* з іменем *Текст* у папці *Навчальні проекти* власної структури папок.
3. Підключіть до файла модуль віконного графічного інтерфейсу користувача, створіть об'єкти: вікно, два текстові надписи, що відображають зміст висловлювання та його автора. Скористайтесь командами, як на малюнку 16.2, *б*, або завантажте заготовку фрагмента коду програми з файла *Вікно* папки *Програмування*.
4. Розмістіть напис і команду запуску опрацювання події програми на головній формі. Наприклад, для виведення у вікні повідомлення тексту цитати потрібно ввести такий програмний код:

```
label1 = tkinter.Label(text="Людина, яка у XXI столітті
не вмітиме користуватися комп'ютером, буде подібна
до людини XX століття, що не вміє ні писати, ні читати.")
label1.pack()
```

5. Запишіть команди створення напису та запуску опрацювання події для виведення у вікні автора висловлювання.
6. Запустіть файл програми на виконання. Переконайтеся, що на екрані ви отримали потрібне вікно із цитатою.
7. Завершіть роботу із середовищем програмування. Закрийте всі вікна програм.

3. Як працювати з проектами у середовищі програмування *Lazarus*?

Ви знаєте, що файл, створений у середовищі *Скретч*, називається **проектом**. Проект у *Скретчі* зберігається в одному файлі. У ньому зберігаються програми для кожного з виконавців, образи цих виконавців, фони сцени, звуки.

Із проектами працюють не тільки в навчальних середовищах складання та виконання алгоритмів.

Проект — це набір файлів, з якими користувач працює під час створення прикладної програми в об'єктно-орієнтованому середовищі програмування.

До проекту входять як стандартні файли, так і файли, що створюються користувачем. У файлах зберігаються окремі екранні форми та модулі. Кожний проект повинен мати ім'я.

Крім того, під проектом розуміють саму прикладну програму в процесі її створення.

Проект, створений у середовищі *Lazarus*, на відміну від середовища *Скретч*, містить декілька файлів:

- *project1.exe*, за допомогою якого запускають проект на виконання;
- *unit1* для збереження програмного коду проекту, написаного мовою програмування *Free Pascal* (файл *unit1.pas*);
- *unit.lfm*, що містить дані, отримані в процесі проектування форми;
- інші файли та папки.

Мал. 16.4

Папка проекту може містити папки та файли, зокрема, як на малюнку 16.4.

У середовищі *Lazarus* можна створювати нові проекти чи використовувати шаблони проектів, відкривати створені проекти, закривати та зберігати їх, змінювати структуру складових проекту тощо. Ці дії виконують за допомогою відповідних вказівок меню *Проект* (мал. 16.5).

Після вибору відповідної вказівки з меню у вікнах, що відкриваються, вказують необхідні параметри та задають їх значення. Наприклад, за допомогою вказівки *Проект/Новий проект* відкривається вікно, у якому вказують тип проекту: програма з графічним інтерфейсом, найпростіша програма тощо (мал. 16.6).

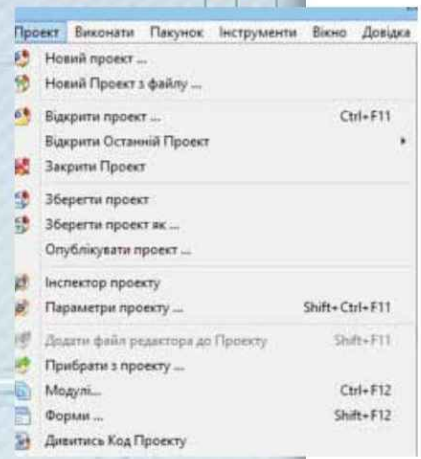
При завантаженні середовища *Lazarus* за замовчуванням автоматично завантажується останній проект, з яким працювали в цьому середовищі. Щоб цього не відбувалось, у меню *Інструменти* слід обрати вказівку *Параметри*, у вікні *Параметри IDE* — зняти позначку *При старті відкрити останній проект* (мал. 16.7) та завершити обрані налаштування натисненням кнопки *Гаразд*.

Завантажений або створений проект можна виконати. Для цього використовують інструмент *Виконати* , розташований на панелі інструментів або в меню *Виконати*.

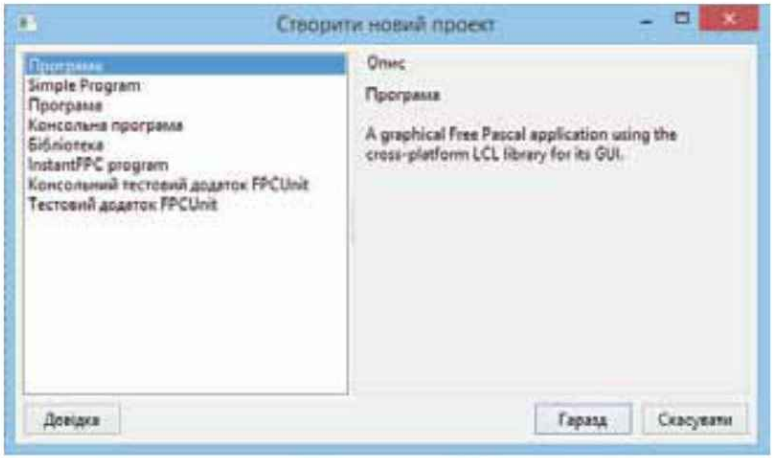
Після запуску проекту на виконання у вікні *Повідомлення* відображаються відомості про хід завантаження та компіляції проекту (мал. 16.8).

Після компіляції проекту користувач може працювати з **екранною формою**, згорнувши або закривши вікно середовища.

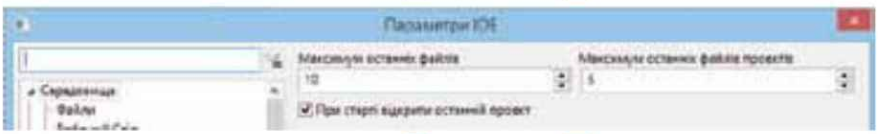
Екранна форма — це вікно, що містить візуальні графічні елементи або об'єкти управління, такі як меню, кнопки. Кожний такий об'єкт може мати зв'язаний із ним функціональний програмний код, який оформля-



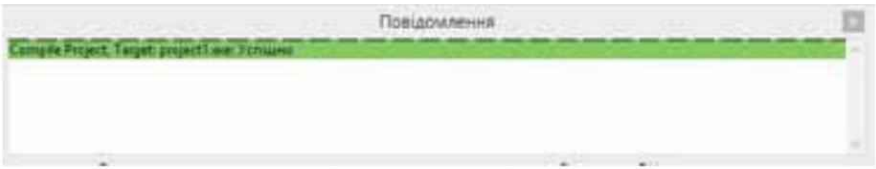
Мал. 16.5



Мал. 16.6



Мал. 16.7



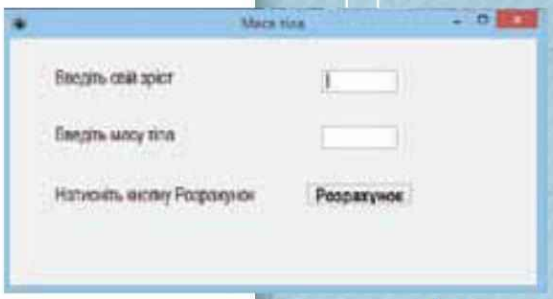
Мал. 16.8

ється у вигляді процедури опрацювання події. Подійні процедури можна розмістити лише в межах програмного коду, пов'язаного з однією конкретною екранною формою. Після запуску програми на виконання для кожної екранної форми створюється файл із розширенням *lfm*, у якому збережені відомості, отримані в процесі проектування форми.

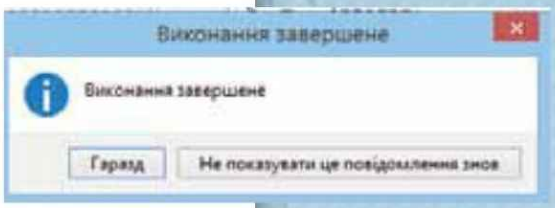
ДІЄМО

Вправа 2. Завантаження та виконання проекту.
Завдання. Завантажте та виконайте проект *Маса тіла*, збережений у папці *Навчальні проекти*. Введіть дані, необхідні для виконання проекту, та проаналізуйте результат.

1. Відкрийте середовище програмування *Lazarus*.
2. У меню *Проект* оберіть вказівку *Відкрити проект*.
3. Перейдіть до папки *Навчальні проекти*. У папці проекту *Маса тіла* відкрийте файл проекту *project1.lpi*.
4. Запустіть проект на виконання. Введіть у пропоновані поля свій зріст у сантиметрах і масу в кілограмах (мал. 16.9).
5. Натисніть кнопку *Розрахунок*, щоб перевірити показники співвідношення своєї ваги та зросту. Проаналізуйте отриманий результат.
6. Закрийте вікно форми проекту. У вікні повідомлення про завершення виконання проекту натисніть кнопку *Гаразд* (мал. 16.10).
7. Завершіть роботу із проектом. Для цього в меню *Проект* оберіть вказівку *Закрити проект*.
8. Завершіть роботу із середовищем. Для цього у вікні *Майстер проектів* оберіть Вийти з Lazarus.



Мал. 16.9



Мал. 16.10

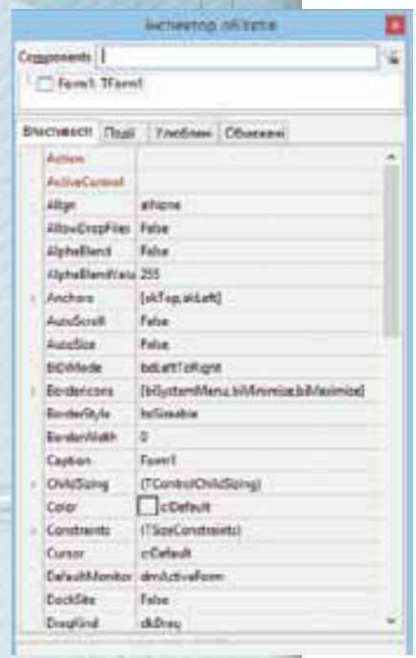
4. Як працювати з екранною формою в середовищі програмування *Lazarus*?

Екранну форму можна створити у вікні дизайнера форми проекту, який містить сукупність створених користувачем об'єктів. У той же час у вікні інспектора об'єктів відображатиметься список властивостей кожного об'єкта, створеного користувачем, та форми проекту.

На початку створення екранної форми дизайнер форми проекту виглядає як порожня робоча область, на якій користувач може розміщувати різні **об'єкти** або **елементи управління**: текстові написи, кнопки, перемикачі, малюнки тощо, подібно до полотна для художника, на якому той малює предмети (мал. 16.11).

Мал. 16.11

Мал. 16.12



Мал. 16.13

Екранна форма також є об'єктом, отже, має властивості та ім'я, вона «може реагувати» на події з деякого списку (мал. 16.12).

За замовчуванням екранна форма має ім'я `Form1`. За необхідності можна відкрити додаткові форми, що матимуть імена `Form2`, `Form3` і т. д. Усі ці форми будуть рівноправними та незалежними одна від одної.

Додавання нової форми виконується за допомогою відповідного розділу меню *Файл/Нова форма* чи за допомогою інструмента *Нова форма* , розташованого на панелі інструментів *Швидкі клавіші IDE*. Щоб переглянути список усіх форм проекту, можна скористатись інструментом *Показувати форми* , який розташований на тій же панелі інструментів.

Щоб змінити властивості форми, наприклад ім'я, використовують таблицю вкладки *Властивості* інспектора об'єктів (мал. 16.13).

Усі властивості об'єктів поділяють на прості та складені. Прості властивості мають одне значення. Наприклад, властивість `Caption` (*Заголовок*) визначається рядком символів, що можуть замінити ім'я форми `Form1`. Властивості `Height` (*Висота*) і `Width` (*Ширина*) — числовим значенням. Властивість, яка визначає, чи буде розмір вікна автоматично добиратись відносно вмісту `AutoSize` (*Авторозмір*) — значеннями `True` (*Істина*) і `False` (*Хибність*). Властивість `Position` (*Розташування*) обирається зі списку можливих. Складені властивості мають кілька значень. Наприклад, властивість `Font` (*Шрифт*) має кілька параметрів форматування: накреслення, розмір, колір символів тощо, значення яких можна змінити у вікні *Шрифт* (мал. 16.14). Вікно для зміни складених властивостей відкривається за допомогою кнопки , що розташована в правій частині рядка обраної властивості.

5. Як у середовищі *Lazarus* розробити прикладну програму?

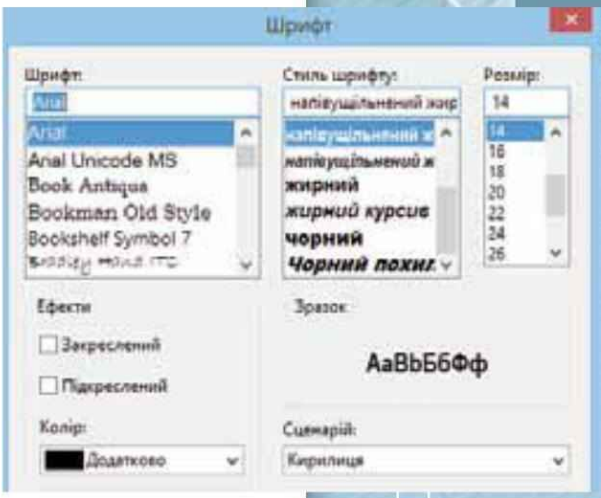
Розробка прикладної програми в середовищі *Lazarus* складається з декількох етапів:

1. Аналіз та планування виконання завдання.
2. Підготовка проекту.
3. Розміщення компонентів інтерфейсу користувача на формі проекту.
4. Написання програми опрацювання подій.
5. Тестування та налагодження програми.

На першому етапі, крім визначення вхідних даних і вихідних результатів, необхідно побудувати інформаційну модель розв'язку завдання,

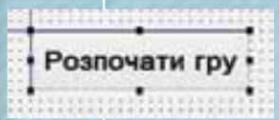
визначити методи розв'язування та скласти відповідні алгоритми.

На етапі підготовки проекту варто створити окрему папку для нього. Після запуску середовища програмування *Lazarus* слід створити новий проект — програму з підтримкою графічного інтерфейсу, зберегти його у створену папку, перевірити успішність компіляції та запуску нового порожнього проекту — це можна зробити також, використовуючи клавішу *F9*. Далі — закрити запущений порожній проект і продовжити його створення. Швидко зберігати всі проміжні зміни можна за допомогою комбінації клавіш *Ctrl+S*.



Мал. 16.14

При створенні графічного інтерфейсу програми слід розмістити компоненти цього інтерфейсу на формі та визначити їх властивості. Щоб об'єкти були розміщені на формі на одному рівні по горизонталі чи вертикалі, використовують автовирівнювання, яке показує розміщення одних компонентів по горизонталі та по вертикалі відносно інших лініями вирівнювання (мал. 16.15).



Мал. 16.15

Програми опрацювання подій створюються у вікні редактора коду автоматично системою або складаються вручну засобами мови програмування *Free Pascal*.

Створений проект запускають на виконання. Якщо код програми не містить синтаксичних помилок і програма успішно відкомпільована, буде відображено екранну форму. Слід перевірити, чи всі об'єкти екранної форми «реагують» на події так, як це передбачено завданням проекту. За наявності невідповідності поставленим завданням необхідно внести зміни до коду програми чи змінити властивості об'єктів.

ДІЄМО

Вправа 3. Створення проекту в середовищі *Lazarus*.

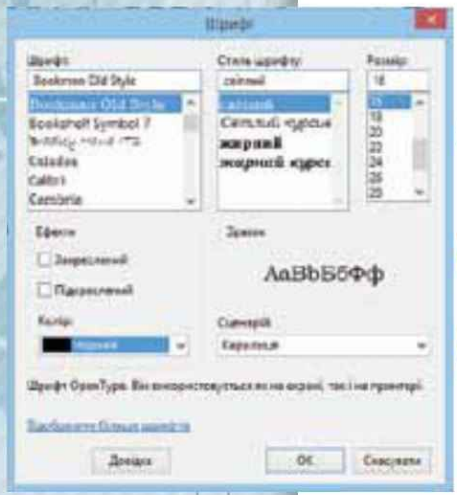
Завдання. Створити проект, після запуску якого у вікні *Перша програма* виводиться повідомлення *Ура! Запрацювало!*

1. Сплануйте проект. Передбачте, який текст має з'являтись у рядку заголовка екранної форми, а який — у її робочій області. Намалюйте схему розміщення компонентів графічного інтерфейсу.
2. Виконайте дії підготовчого етапу розробки проекту. Створіть папку *Перша програма*.
3. Після запуску середовища *Lazarus,* збереження проекту та його компіляції перейдіть у вікно дизайнера форми.
4. У таблиці властивостей форми оберіть властивість `Caption` (*Підпис*), надайте значення цієї властивості — *Перша програма*.
5. На панелі компонентів *Standart* (*Стандартна*) оберіть інструмент Abc , за допомогою якого створюється напис на формі.
6. Клацніть у місці форми, де ви передбачаєте розмістити об'єкт *напис*.
7. У таблиці властивостей напису оберіть властивість `Caption` (*Підпис*), надайте значення цієї властивості — *Ура! Запрацювало!*. Перевірте, чи змінився напис *Label1* на введений текст.

8. Змініть значення властивості `Font` (*Шрифт*) так, як на малюнку 16.16.
9. Збережіть зміни у проекті. Запустіть його на виконання. Згорніть усі вікна середовища *Lazarus*. За потреби відлагодьте проект. Закрийте всі вікна.

ОБГОВОРЮЄМО

Обговоріть запитання, що містяться у файлі *Тема 16*, який зберігається у папці *Обговорюємо*. Також відкрити ці запитання можна за допомогою QR-коду.



Мал. 16.16

ПРАЦЮЄМО В ПАРАХ

1. Обговоріть призначення основних файлів, з яких складається проект (мал. 16.4). Поміркуйте, чому деякі файли мають однакове ім'я, але різні розширення. Яке їх можливе призначення у проекті, створеному в середовищі *Lazarus*?
2. Визначте три аргументи, які підтверджують, що навчальне середовище програмування *Скретч* підтримує подійно-орієнтоване програмування.
3. Запустіть на виконання проект *Чарівна скриня*, збережений у папці *Проект Скриня* папки *Навчальні проекти* (мал. 16.17). Обговоріть, які компоненти графічного інтерфейсу могли бути використані при його складанні, які їх основні властивості. Переконайтесь у правильності своїх здогадок, відкривши проект у середовищі *Lazarus*.

Мал. 16.17

ПРАЦЮЄМО САМОСТІЙНО

1. Ваш молодший брат любить довго засиджуватися за комп'ютером. Створіть програму в середовищі *PyCharm*, після запуску якої з'являтиметься вікно з повідомленням про необхідність дотримання часу роботи за комп'ютером. Створіть аналогічну програму в середовищі *Lazarus*. Визначте:
   1) скільки часу ви витратили на розробку кожної із програм, поміркуйте, чому зафіксовані значення часу розробки програми є різними;
   2) вікно якої екранної форми проекту більше б зацікавило молодшого брата і чому.
2. Ви хочете застосувати власний комп'ютер для планування діяльності своєї родини. Оберіть середовище програмування, сплануйте та створіть прикладну програму, після запуску якої у вікні з'являтиметься повідомлення, актуальне для вашої родини, наприклад, нагадування про день народження, важливу подію, заплановане завдання тощо.
3. Оберіть середовище програмування *Lazarus* або

ЦІКАВІ ДЖЕРЕЛА

Основи програмування мовою *Python*: *https://www.thinkful.com/learn/intro-to-python-tutorial/*
Вікі-сторінка довідкових матеріалів по середовищу *Lazarus*: *http://wiki.freepascal.org/Lazarus_Tutorial/ru*

PyCharm та створіть у ньому проект, після запуску якого з'являтиметься вікно з описом понять цього уроку: проект, екранна форма, інтерфейс користувача.

4. Створіть програму, після запуску якої на екранній формі буде відображено календар на поточний місяць, наприклад, на листопад 2016 р. (мал. 16.18). Скористайтесь підказкою: компоненти графічного інтерфейсу, розміщені на формі, можна копіювати, наприклад, за допомогою вказівок контекстного меню. Після вставки створюється новий компонент, що «наслідує» властивості попереднього, які можна змінити.

Мал. 16.18

17. ПРАКТИЧНА РОБОТА 8

СТВОРЕННЯ ОБ'ЄКТНО-ОРІЄНТОВАНОЇ ПРОГРАМИ, ЩО ВІДОБРАЖАЄ ВІКНО ПОВІДОМЛЕННЯ

ПРИГАДАЙТЕ

- як розміщувати текстові написи на екранній формі проекту;
- як змінювати властивості об'єктів програми;
- як створювати проекти в середовищах програмування *PyCharm* та *Lazarus*.

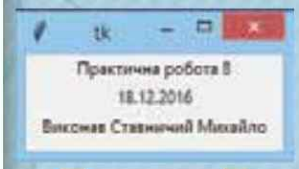
СТВОРІТЬ

У власній структурі папок створіть папку *Практична робота 8.*

ПАМ'ЯТАЙТЕ

Під час виконання практичних завдань пам'ятайте про правила безпеки життєдіяльності при роботі з комп'ютером!

Завдання 1. Повідомлення про виконання практичної роботи (*9 балів*)
Складіть програму *Практична_робота* в середовищі *PyCharm*, за допомогою якої у вікні буде виводитись повідомлення про номер практичної роботи, дату її виконання, прізвище та ім'я виконавця за зразком (мал. 17.1).



Мал. 17.1

Завдання 2. Кислоти (*9 балів*)
Розподіліть назви кислот у вікні програмного проекту *Кислоти*, збереженого в папці *Програмування,* на дві групи. Змініть властивості написів так, щоб значення параметрів форматування символів у назвах груп були однаковими.

Завдання 3. Словосполучення (*12 балів*)
Складіть програму, яка у вікні із заголовком *Словосполучення* буде виводити нагадування про відомі з уроків української мови види зв'язку слів у словосполученні (мал. 17.2).

Мал. 17.2

18. ВЛАСТИВОСТІ ТА МЕТОДИ ЕКРАННОЇ ФОРМИ І ЕЛЕМЕНТІВ УПРАВЛІННЯ

ПРИГАДАЙТЕ:

- які дії можна виконати з елементом управління *кнопка* у програмах із графічним інтерфейсом;
- для чого використовуються кнопки управління вікном в ОС *Windows*;
- як у середовищі *Скретч* можна створити об'єкт *кнопка*;
- за допомогою яких команд у середовищі *Скретч* можна передати дію одного об'єкта іншому.

ВИ ДІЗНАЄТЕСЯ:

- як змінити значення властивостей об'єктів у середовищі програмування *Lazarus*;
- як об'єкти можуть «реагувати» на події;
- як у проекті використовують елемент управління *кнопка*;
- як виконувати дії з об'єктами за допомогою методів;
- як у середовищах програмування можна використовувати вікна повідомлень.

ВИВЧАЄМО

1. Як змінити значення властивостей об'єктів у середовищі програмування *Lazarus*?

Усі елементи управління, які можна розміщувати на формі проекту, як і сама форма, є об'єктами. Ви вже вмієте встановлювати властивості об'єкта, зокрема напису та форми, **статичним** способом. Це означає, що значення властивостей об'єктів встановлюються до запуску програми на виконання. Користувач спочатку виділяє у вікні дизайнера форми об'єкт, значення властивості якого необхідно змінити, а потім у вікні *Інспектор об'єктів* у таблиці властивостей знаходить назву відповідної властивості та праворуч від цієї назви вибирає або вводить із клавіатури потрібне значення.

Для зміни значень властивостей можна застосувати й інший спосіб — **динамічний**, коли значення властивостей можна змінити в процесі виконання програми за допомогою команди надання значень — **присвоювання**. Мовою програмування *Free Pascal*, яка підтримується в середовищі *Lazarus*, таку команду позначають «:=», а мовою *Python* — просто «=». Зліва від такого оператора записується ім'я властивості, а справа — значення, якого необхідно надати. Такі команди записують у програмному коді, і значення властивості буде змінюватися тільки після запуску програми на виконання. Для того щоб при написанні програмного коду вказати, що значення властивості слід змінити саме для обраного об'єкта, необхідно, крім назви властивості в команді про надання значення, ще вказувати ім'я об'єкта, до якого такі зміни будуть застосовані. Якщо проект складається з кількох екранних форм, слід вказати також ім'я форми, на якій розміщений такий об'єкт. Тому для звернення до властивостей об'єктів у програмному коді використовують такий спосіб запису:

Ім'я форми. Ім'я об'єкта. Ім'я властивості

Цікаво

У середовищі програмування *PyCharm* використовують динамічний спосіб зміни властивостей об'єктів. Так, за допомогою команди

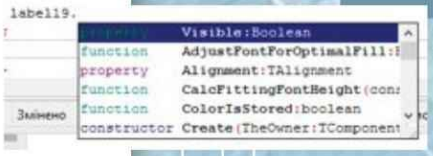
```
label = tkinter.Label (text="Hello World!")
```

не тільки створюють напис у вікні, а й вказують текст, який має бути відображений у цьому написі до запуску програми на виконання.

Якщо проект містить лише одну форму або розглядаються об'єкти поточної форми, то вказувати ім'я форми не обов'язково.

Так, наприклад, щоб у проекті, який містить лише одну форму, об'єкт `Label1` (*Напис1*) відображався на формі, необхідно його властивості `Visible` (*Видимість*) присвоїти значення `True`. Для цього в коді програми можна записати команду:

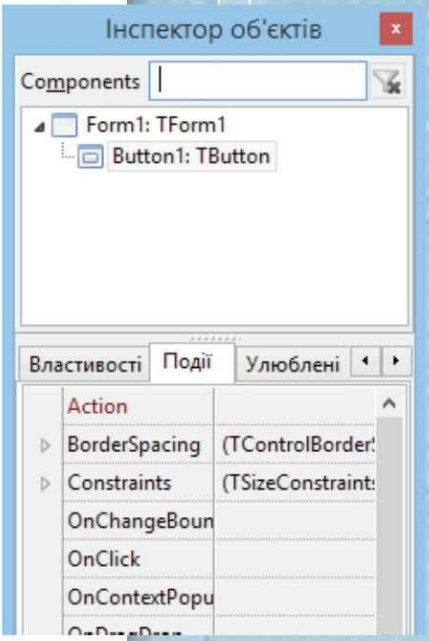
```
Label1.Visible := True;
```



Мал. 18.1

Властивість `Visible` (*Видимість*) може набувати одного з двох значень: `True` — тоді об'єкт буде відображено, і `False` — у цьому разі отримаємо невидимий, або «прихований» об'єкт.

У вікні редактора коду середовища програмування *Lazarus* після введення імені об'єкта і крапки відкривається список доступних властивостей та дій, які можна описати для об'єкта (мал. 18.1). Щоб додати в код ім'я потрібної властивості, достатньо обрати її мишею зі списку.



Мал. 18.2

2. Як об'єкти можуть «реагувати» на події?

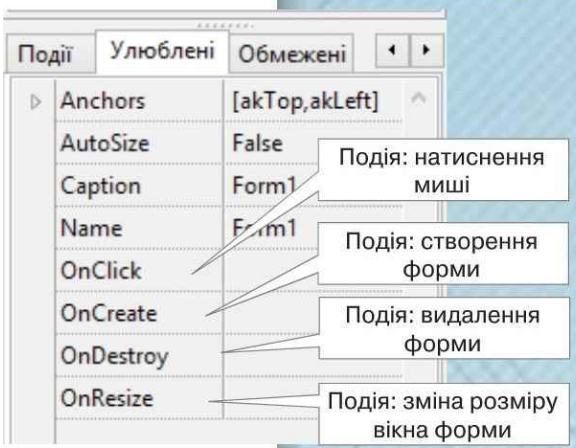
Під час виконання програми об'єкти можуть «реагувати» на деякі **події**, зокрема, дії користувача: клацання мишею на об'єкті, натиснення на клавіатурі деякої клавіші чи комбінації клавіш, вибір деякої вказівки меню, зміну розмірів вікна тощо. Тобто в результаті настання деякої події, що передбачена для об'єкта, виконується певний набір команд. Наприклад, під час роботи з офісними програмами при першому збереженні файла користувач може обрати на панелі інструментів кнопку *Зберегти*, або натиснути на клавіатурі комбінацію клавіш *Ctrl* + *S*, або в меню *Файл* обрати вказівку *Зберегти*. При настанні будь-якої з цих подій буде виконуватися дія: відкриється вікно *Зберегти як*, у якому можна обрати ім'я файла та папку, до якої цей файл слід зберегти.

Щоб описати дії, які мають виконуватися при настанні деякої події для об'єкта, необхідно написати програмний код — підпрограму, яка в середовищі *Lazarus* розпочинається зі службового слова `procedure`.

Список усіх доступних подій, які можуть опрацьовуватися, для кожного об'єкта в середовищі програмування *Lazarus* подається в таблиці на вкладці *Події* вікна *Інспектор об'єктів* (мал. 18.2).

Список подій, які найчастіше застосовують при складанні проектів з обраним об'єктом, відображається в таблиці на вкладці *Улюблені*. Наприклад, список таких подій для форми подано на малюнку 18.3.

Якщо двічі клацнути у клітинці таблиці справа від обраної події, то у вікні редактора коду



Мал. 18.3

```
procedure TForm1.FormCreate(Sender: TObject);
begin

end;
```

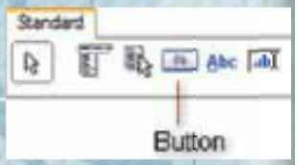
Ім'я об'єкта

Ім'я події

Початок програмного коду

Кінець програмного коду

Мал. 18.4



Мал. 18.5

з'являється фрагмент програмного коду для опису реакції на подію (мал. 18.4).

Набір команд, який використовується для опису реакції на конкретну подію й записується за правилами мови програмування, називається **процедурою**. Процедура розпочинається зі слова `procedure`, обов'язково містить пару службових слів, які обмежують список команд у процедурі: `begin` — початок програмного коду, `end` — кінець. Після службового слова `begin` символ «;» не ставиться, після `end`, так само як після кожного рядка програми, слід поставити «;».

У середовищі програмування *PyCharm* властивості об'єктів та процедури записуються відповідними командами у програмному коді.

3. Як у проекті використовують елемент управління *кнопка*?

Кнопки, які створюють у середовищі програмування *Lazarus* за допомогою компонента `Button` (*Кнопка*) (мал. 18.5), використовують, щоб під час виконання програми після їх натиснення виконувався деякий набір команд.

Кнопки мають властивості, призначення яких є аналогічним до властивостей інших об'єктів (`Caption` — підпис, `Enabled` — увімкнення, `Font` — шрифт, `Height` — висота, `Left` — відступ від краю екранної форми ліворуч, `Name` — ім'я, `Visible` — видимість, `Width` — ширина) та ін.

З об'єктом *кнопка* найчастіше пов'язують подію `OnClick`. Щоб перейти до редактора програмного коду, у якому потрібно вказати команди, що будуть виконуватися після запуску проекту та натиснення кнопки, можна двічі клацнути на кнопці у вікні дизайнера форми. У вікні редактора коду додається процедура опрацювання події — *натиснення кнопки* (мал. 18.6).

```
procedure TForm1.Button1Click(Sender: TObject);
begin

end;
```

Ім'я події натиснення кнопки *Button1*

Початок програмного коду

Кінець програмного коду

Мал. 18.6

Команди, які необхідно виконати у відповідь на натиснення кнопки, *слід записати між* службовими словами `begin` та `end`.

ДІЄМО

Вправа 1. Прості та складені числа.

Завдання. Змініть проект *Числа*, який збережений у папці *Програмування*, так, щоб після натиснення кнопки *Складені числа* на екрані відображалися складені натуральні числа перших двох десятків, а після натиснення кнопки *Прості числа* — прості.

1. Скопіюйте папку з проектом *Числа*, який збережений у папці *Програмування*, у папку *Навчальні проекти* власної структури папок.
2. Завантажте середовище програмування *Lazarus*. У меню *Проект* оберіть вказівку *Відкрити проект.*

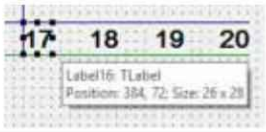
3. Перейдіть до папки *Навчальні проекти*. У папці проекту *Числа* відкрийте файл проекту *project1.lpi*.
4. Запустіть проект на виконання. Перевірте, які дії виконуються в програмі після натиснення кнопки *Складені числа*.
5. Переконайтеся, що з кнопкою *Прості числа* не пов'язана жодна подія у програмі й при натисненні цієї кнопки ніякі дії не відбуваються.
6. Зупиніть виконання програми, натиснувши кнопку *Завершити* .
7. Проаналізуйте програмний код, який викликається подією: натиснуто об'єкт `Button1` (*Кнопка1*) (мал. 18.7).
8. Перейдіть у вікно дизайнера форм. Двічі клацніть на об'єкті `Button2` (*Кнопка2*), значення властивості `Caption` якого — *Прості числа*.
9. Перейдіть у вікно редактора коду. Створіть програмний код подібно до відповідного коду для об'єкта `Button1` (*Кнопка1*) (мал. 18.7).
10. Збережіть зміни у проекті, за потреби відлагодьте його. Запустіть проект на виконання.
11. Завершіть роботу з проектом і середовищем програмування.

```
procedure TForm1.Button1Click(Sender: TObject)
begin
    label1.Visible:= False;
    label2.Visible:= False;
    label3.Visible:= True;
    label4.Visible:= False;
    label5.Visible:= True;
    label6.Visible:= False;
    label7.Visible:= True;
    label8.Visible:= True;
    label9.Visible:= True;
    label10.Visible:= False;
    label11.Visible:= True;
    label12.Visible:= False;
    label13.Visible:= True;
    label14.Visible:= True;
    label15.Visible:= True;
    label16.Visible:= False;
    label17.Visible:= True;
    label18.Visible:= False;
    label19.Visible:= True;
end;
```

У написі, що містить число 2, властивість `Visible` набуває значення `False`. Тому напис **не з'являється** на екранній формі

У написі, що містить число 4, властивість `Visible` набуває значення `True`. Тому напис **з'являється** на екранній формі

Ім'я та інші властивості напису, які можна використати в програмному коді, «підсвічуються» при наведенні на нього мишею



Мал. 18.7

Вправа 2. Чарівні кнопки.

Завдання. Розробіть проект, у якому під час натиснення кнопки *Зліва-направо* або *Справа-наліво* з поданих фрагментів слів «КА», «БАН» утворюється слово у вказаному порядку.

1. У папці *Навчальні проекти* власної структури папок створіть папку *Чарівні кнопки*.
2. Запустіть середовище *Lazarus*, створіть новий проект. Змініть такі властивості об'єкта `Form1` (табл.18.1).
3. У вікні дизайнера форми `Form1` додайте на форму об'єкти, які мають вказані властивості (табл. 18.2):

Таблиця 18.1

Властивість	Значення властивості
`Caption`	Чарівні кнопки
`Font`	*Шрифт*: Book Antiqua, *Стиль шрифту*: звичайний, *Розмір*: 16

Таблиця 18.2

Об'єкт	Властивість	Значення властивості
Label1	Caption	КА
	Font	*Шрифт*: Book Antiqua, *Стиль шрифту*: жирний курсив, *Розмір*: 20
Label2	Caption	БАН
	Font	*Шрифт*: Book Antiqua, *Стиль шрифту*: жирний курсив, *Розмір*: 20
Label3	Caption	СЛОВО
	Font	*Шрифт*: Book Antiqua, *Стиль шрифту*: жирний курсив, *Розмір*: 20
Button1	Caption	Зліва-направо
	Font	*Шрифт*: Book Antiqua, *Стиль шрифту*: звичайний, *Розмір*: 20
Button2	Caption	Справа-наліво
	Font	*Шрифт*: Book Antiqua, *Стиль шрифту*: звичайний, *Розмір*: 20

Цікаво

У різних середовищах назви властивостей об'єктів можуть бути подані синонімами. У деяких версіях офісних програм, зокрема, в середовищі текстового процесора, під час форматування тексту властивість, за допомогою якої символи можна подати курсивом чи напівжирними, називають *Накресленням*. У середовищі програмування *Lazarus* така властивість має назву *Стиль шрифту*.

4. Двічі клацніть на кнопці `Button1`. У редакторі коду запишіть команду, за якою у властивість `Caption` напису `Label3` буде надано значення властивості `Caption` напису `Label1`, до якого додане значення властивості `Caption` напису `Label2`:

```
Label3.Caption:= Label1.Caption+Label2.Caption;
```

5. Аналогічно додайте програмний код до процедури, яка буде опрацьовувати подію — *натиснення кнопки* `Button2`. Візьміть до уваги, що в цьому разі напис має формуватися в іншій послідовності.
6. Збережіть проект й усі файли, які його складають. Запустіть проект на виконання. Перевірте, чи виконуються вказані в завданні дії під час натиснення кожної з кнопок. За потреби виправте помилки.
7. Завершіть роботу із проектом і середовищем програмування.

4. Як виконувати дії з об'єктами за допомогою методів?

Крім процедур, які складає програміст для опрацювання подій, що відбуваються у зв'язку з діями користувача, у середовищі програмування *Lazarus* міститься велика кількість вбудованих процедур і функцій. Деякі з вбудованих процедур є **методами** об'єктів, за допомогою яких можна виконувати дії з об'єктами. Для різних об'єктів передбачені різні методи, їх кількість і призначення залежать від конкретного об'єкта.

Метод — це фрагмент програмного коду, який вбудований в об'єкт та передбачає виконання деяких дій з ним.

Метод має ім'я й уже містить набір команд, які виконуються, якщо викликати метод для об'єкта. Запис команди для виклику методу складається з імені об'єкта та імені методу, що розділені крапкою:

```
Ім'я об'єкта.Ім'я методу;
```

Ім'я методу, як і ім'я властивості, можна ввести з клавіатури або обрати зі списку, який відкривається, якщо в коді програми записати ім'я об'єкта і крапку (див. мал. 18.1). У такому списку відображені саме ті методи, які можна викликати для обраного об'єкта.

Якщо програма містить кілька форм, то за замовчуванням відображується лише головна форма. Для відображення інших форм проекту використовують метод `Show` або `ShowModal`, наприклад:

```
Form2.ShowModal;
```

Відмінність цих методів полягає в тому, що перший викликають для відображення форми у звичайному режимі, а другий — у так званому **модальному** режимі: після відображення вказаної форми всі інші форми стануть недоступними; щоб перейти до іншої форми, необхідно закрити поточну форму.

Для об'єкта *форма* найчастіше використовують такі методи:

- `Show` — відбразити форму в звичайному режимі;
- `ShowModal` — відобразити форму в модальному режимі;
- `Hide` — зробити форму невидимою (приховати її);
- `Close` — закрити форму.

Деякі методи можна викликати для різних об'єктів, а інші — лише для об'єктів певного типу. Наприклад, методи `Show` і `Hide` можна викликати для форми, напису, кнопки та інших об'єктів, а метод `Close` — лише для форми.

Якщо у проекті використовують декілька екранних форм, то у програмному коді слід задати команду `uses`, за допомогою якої підключаються програмні модулі інших форм. Після команди `uses` подають список імен файлів екранних форм, які будуть підключені. Наприклад, якщо буде використана форма, збережена у файлі `Unit2`, то у програмному коді вказують:

```
uses Unit2;
```

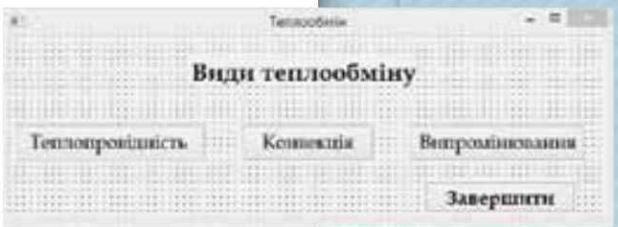
Виклик методів для таких «зовнішніх» програм буде виглядати так:

```
Ім'я файла.Ім'я об'єкта.Ім'я методу;
```

ДІЄМО

Вправа 3. Довідник із фізики.

Завдання. Розробіть проект, у якому головна екранна форма матиме вигляд, як на малюнку 18.8.



Мал. 18.8

Для кожної з кнопок форми налаштуйте опрацювання події *натиснення кнопки* так, щоб натиснення кнопки *Завершити* приводило до закриття вікна форми. А за допомогою кнопок *Теплопровідність*, *Конвекція* та *Випромінювання* — викликалися вікна з поясненням відповідного виду теплообміну так, що доки вікно з поясненням не закрите, до головної екранної форми перейти не можна.

1. У папці *Навчальні проекти* власної структури папок створіть папку *Фізика*.
2. Запустіть середовище *Lazarus*, створіть новий проект. Змініть властивості об'єкта `Form1` (табл. 18.3):
3. У вікні дизайнера форми `Form1` додайте об'єкти, які мають такі властивості (табл. 18.4). Для об'єктів `Button1`—`Button4` задайте значення властивості `Font`: *Шрифт*: Book Antiqua, *Стиль шрифту*: звичайний, *Розмір*: 16
4. Додайте до проекту форму `Form2`, розмір якої відповідає розміру головної екранної форми. На екранній формі `Form2` розмістіть два написи, надавши для першого значення властивості

Таблиця 18.3

Властивість	Значення властивості
`Caption`	Теплопровідність
`Font`	*Шрифт*: Book Antiqua, *Стиль шрифту*: звичайний, *Розмір*: 16

Таблиця 18.4

Об'єкт	Властивість	Значення властивості
Label1	Caption	Види теплообміну
	Font	*Шрифт*: Book Antiqua, *Стиль шрифту*: жирний, *Розмір*: 20
Button1	Caption	Теплопровідність
Button2	Caption	Конвекція
Button3	Caption	Випромінювання
Button4	Caption	Завершити

`Caption`: *Теплопровідність — перенесення енергії від більш нагрітих частин тіла до менш нагрітих,* для другого — *внаслідок теплового руху* та *взаємодії частинок.* Доберіть значення властивостей форми й написів так, щоб текст на екранній формі можна було прочитати.

5. Додайте до проекту форми `Form3` та `Form4`, які містять написи: *Конвекція — перенесення енергії струменями рідини або газу* та *Випромінювання — перенесення енергії за допомогою електромагнітних хвиль*. Доберіть значення властивостей форми й написів, як для об'єкта `Form2`.

6. Розмістіть вікна додаткових екранних форм під головною формою — одне за одним.

7. Збережіть проект і всі файли, які його складають.

8. Перейдіть до вкладки редактора коду головної екранної форми (мал. 18.10).

Введіть текст у редакторі коду після вказівки `Implementation` (*Впровадження*):

```
uses Unit2, Unit3, Unit4;
```

9. Перейдіть у вікно дизайнера форми `Form1`. Двічі клацніть на кнопці *Теплопровідність*. У вікні редактора коду додайте виклик методу `ShowModal` для форми `Form2` (мал. 18.10): показати форму, яка збережена у файлі `Unit2`, так, щоб інші вікна були недоступними.

10. Виконайте аналогічні до п. 9 дії для кнопок *Конвекція* та *Випромінювання,* за якими будуть відкриватись відповідні екранні форми.

11. Додайте метод

```
Form1.Close;
```

який викликається натисненням кнопки *Завершити*.

12. Запустіть проект на виконання. Перевірте дії, які виконуються під час настання подій для об'єктів екранних форм.

13. Завершіть роботу з проектом і середовищем.

Мал. 18.9

```
procedure TForm1.Button1Click(Sender: TObject);
begin
      Unit2.Form2.ShowModal;

end;
```

Мал. 18.10

5. Як у середовищах програмування можна використовувати вікна повідомлень?

Для виведення даних в окремому вікні в середовищі програмування *Lazarus* можна скористатись також командою виклику вікна `MessageDlg`, яка має таку структуру:

`MessageDlg(повідомлення, тип_вікна_повідомлення, [список_кнопок], довідка);` де

- `повідомлення` — текст, який буде відображено у вікні повідомлення;
- `тип_вікна_повідомлення` — визначає зовнішній вигляд вікна;
- `список_кнопок` — список даних, поданих через кому, які визначають тип кнопки (не обов'язковий параметр);
- `довідка` — номер вікна довідкової системи, яке буде виведено на екран, якщо натиснути клавішу *F1*. Якщо значення цього параметра дорівнює нулю, то використання довідки не передбачено.

Використовують такі типи вікна повідомлення: `mtInformation` (інформаційне), `mtWarning` (попереджувальне), `mtError` (повідомлення про помилку), `mtConfirmation` (запит на підтвердження), `mtCustom` (звичайне).

`mbYes` — *Так*
`mbNo` — *Ні*
`mbAbort` — *Перервати*
`mbRetry` — *Повторити*
`mbIgnore` — *Пропустити*
`mbHelp` — *Довідка*
`mbOk` — *ОК*
`mbCancel` — *Скасувати*

Мал. 18.11

На вікнах повідомлень можуть бути розміщені кнопки, імена яких подають списком (мал. 18.11).

У середовищі *PyCharm* команда створення вікна повідомлення прописується у програмному коді, яка має вигляд `messagebox` (вікно повідомлення), та яку підключають разом із модулем графічного інтерфейсу `tkinter` (мал. 18.10). Опис події вказують після службового слова `def`.

Наприклад:

```
tkinter.messagebox.showinfo("Program", "Hello Again!")
```

Зверніть увагу, що мовою програмування *Python* під час запису наступної команди в новому рядку важливою є відстань зліва, що формується за допомогою відповідної кількості пропусків. Команди, які мають однаковий рівень по вертикалі, незалежні одна від одної й виконуються послідовно. А для команд, що розпочинаються правіше — кожний наступний рівень вказує, що команди цього рівня виконуються в межах команди вищого рівня.

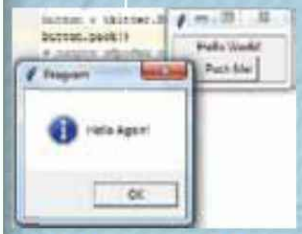
Наприклад, команда

```
tkinter.messagebox.showinfo("Program", "Hello Again!")
```

у коді програми (мал. 18.13) розташована правіше, ніж інші команди. Це означає, що вона виконуватиметься в рамках опрацювання події, що визначена командою вищого рівня:

```
def button_click():
import tkinter
import tkinter.messagebox
# опрацювання події натиснення на кнопку
def button_click():
  tkinter.messagebox.showinfo("Program", "Hello Again!")
main = tkinter.Tk()
# створення текстового напису та його розміщення на
головній формі
label = tkinter.Label(text="Hello World!")
label.pack()
# створення кнопки та її розміщення на головній формі
button = tkinter.Button(main, text="Push Me!", command=button_
click)
button.pack()
# запуск опрацювання подій програми
main.mainloop()
```

Мал. 18.13



Мал. 18.12

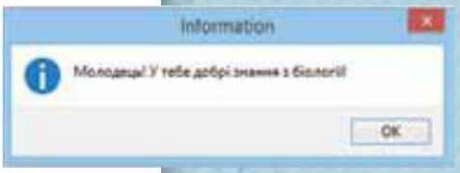
Пезультат виконання програм відображено на малюнку 18.13.

ДІЄМО

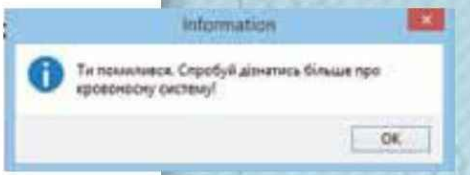
Вправа 4. Тест із біології.

Завдання. Розробіть проект *Тест*, у якому на екранній формі *Кровоносна система* розміщений напис: *Кров виносить із клітин продукти розпаду, що утворюються в результаті їх життєдіяльності,* — та дві кнопки: *Погоджуюсь, Не погоджуюсь*. Якщо натиснута кнопка, що підтверджує істинність твердження, то з'являється вікно, як на малюнку 18.14, а якщо кнопка *Не погоджуюсь* — як на малюнку 18.15.

1. У папці *Навчальні проекти* власної структури папок створіть папку *Тест*.



Мал. 18.14



Мал. 18.15

2. Запустіть середовище *Lazarus*, створіть новий проект. Змініть властивості об'єкта `Form1` (табл. 18.5).
3. У вікні дизайнера форм розмістіть об'єкти (табл. 18.6).

Таблиця 18.5

Властивість	Значення властивості
`Caption`	Кровоносна система
`Font`	*Шрифт*: Book Antiqua, *Стиль шрифту*: звичайний, *Розмір*: 16

Таблиця 18.6

Об'єкт	Властивість	Значення властивості
`Label1`	`Caption`	*Кров виносить із клітин продукти розпаду, що утворюються в результаті їх життєдіяльності*
	`Font`	*Шрифт*: Arial, *Стиль шрифту*: курсив, *Розмір*: 16
`Button1`	`Caption`	Погоджуюсь
	`Font`	*Шрифт*: Book Antiqua, *Стиль шрифту*: звичайний, *Розмір*: 16
`Button2`	`Caption`	Не погоджуюсь
	`Font`	*Шрифт*: Book Antiqua, *Стиль шрифту*: звичайний, *Розмір*: 16

4. Створіть процедуру опрацювання події: натиснуто кнопку *Погоджуюсь*. У вікні редактора коду введіть команду створення вікна повідомлення:

```
MessageDlg('Молодець! У тебе добрі знання з біології!', mtInformation,[mbOk], 0);
```

5. Створіть процедуру опрацювання події: натиснуто кнопку *Не погоджуюсь*. У вікні редактора коду введіть команду створення вікна повідомлення:

```
MessageDlg('Ти помилився. Спробуй дізнатись більше про кровоносну систему!', mtInformation,[mbOk], 0);
```

6. Запустіть проект на виконання. Перевірте, чи відповідають умові завдання дії, пов'язані з об'єктами управління екранної форми. Якщо є помилки, — виправте їх.
7. Завершіть роботу з проектом і середовищем програмування.

ОБГОВОРЮЄМО

Обговоріть запитання, що містяться у файлі *Тема 18*, який зберігається у папці *Обговорюємо*. Також відкрити ці запитання можна за допомогою QR-коду.

ДОСЛІДЖУЄМО

Вправа 5. Обмін написами.
Учень 8 класу розробив проект, у якому на екранній формі відображаються два написи. Після натискання кнопки *Обмін* вміст написів міняється місцями. Учень придумав два способи опису програмного коду для реалізації цієї події. Визначте такі способи та перевірте їх експериментально.

ПРАЦЮЄМО В ПАРАХ

1. Обговоріть переваги й недоліки різних способів зміни значень властивостей об'єктів у середовищі *Lazarus*.
2. Розгляньте екранну форму деякого проекту (мал.18.16). Пограйте у гру «Запитання — відповідь»: один учасник ставить запитання, а інший на нього відповідає. Якщо відповідь неправильна, або в першо-

го учасника закінчилися запитання, то учасники гри міняються ролями. Виграє той, хто останнім назве правильну відповідь. Можливі запитання:

Яким може бути ім'я вказаного об'єкта?
Як змінити його властивість...?
Яка подія може бути пов'язана із вказаним об'єктом?

Мал. 18.16

3. Який вигляд може мати форма проекту, які елементи управління доцільно застосувати та яким чином створити проект, щоб використати його для навчання учнів молодших класів правил безпечної поведінки в Інтернеті? Обговоріть і розробіть дизайн вікна екранної форми, опишіть сценарій майбутнього проекту. Порадьтесь, які команди можна використати у програмному коді. Розподіліть ролі: керівник розробки проекту та програміст. Створіть запланований проект.

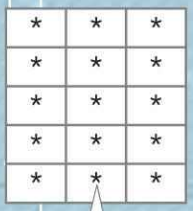
ПРАЦЮЄМО САМОСТІЙНО

1. Розробіть проект *Табло*, у якому на формі відображатимуться написи із символом * (мал. 18.17, *а*) і дві кнопки із цифрами 1 і 2. Після натиснення кнопки із цифрою 1 або 2 на табло зникатимуть деякі із зображених символів, а залишатимуться ті, що відтворюють обрану на кнопці цифру (мал. 18.17 *б*, *в*).

2. У середовищі *Lazarus* розробіть проект *Анаграми*, у якому після натиснення кнопки *Розпочати* з поданих трьох літер, розміщених у написах, на екрані утворюються можливі слова. Наприклад, із літер Л, І, С потрібно отримати: ЛІС, ІСЛ, СЛІ, СІЛ, ЛСІ, ІЛС.

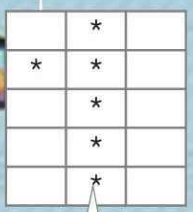
3. Розробіть проект *Тест* із вашого улюбленого навчального предмета та реалізуйте його в середовищі *PyCharm*. Скористайтесь ідеєю проекту тесту з біології й використайте команди, подані на малюнку 18.13.

4. У середовищі *Lazarus* розробіть проект *Моя Україна,* за допомогою якого користувач може отримати відомості про столицю нашої держави та її географічну широту і довготу; державний устрій і дату проголошення незалежності; кількість населення та загальну площу території. Скористайтеся за потреби довідкою з Інтернету. Кількість форм, їх дизайн та об'єкти сплануйте самостійно.



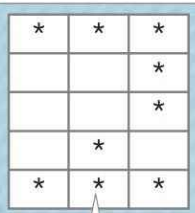
а

Початковий вигляд табло



б

Табло після натиснення кнопки із цифрою 1



в

Табло після натиснення кнопки із цифрою 2

Мал. 18.17

ЦІКАВІ ДЖЕРЕЛА

Матеріал із Вікіпедії про використання методів у програмуванні: *https://uk.wikipedia.org/wiki/Метод_(програмування)*

19. ПРАКТИЧНА РОБОТА 9

СТВОРЕННЯ ПРОГРАМИ З КНОПКАМИ ТА НАПИСАМИ

ПРИГАДАЙТЕ

- як використовувати елементи управління *кнопка* та напис на екранних формах;
- як змінювати значення властивостей об'єктів у програмному коді;
- як використовувати вікна повідомлень.

СТВОРІТЬ

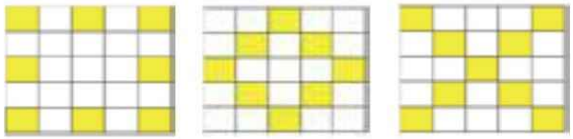
У власній структурі папок створіть папку *Практична робота 9*.

ПАМ'ЯТАЙТЕ

Під час виконання практичних завдань пам'ятайте про правила безпеки життєдіяльності при роботі з комп'ютером!

Завдання 1. Мозаїка (*12 балів*)

Змініть проект *Мозаїка* так, щоб деяке слово, подане на формі написом *Label1*, після натиснення на кнопки *Квадрат*, *Ромб*, *Діагональ* відображались у написах, розміщених у позиціях, що виділені на малюнку жовтим кольором відповідно до форми вибраної фігури. Скористайтеся підказкою щодо номерів написів:



Завдання 2. Подільність чисел (*20 балів*)

Розробіть проект *Подільність чисел*, у якому з перших 10 натуральних чисел, розміщених на екранній формі, після натиснення кнопки з обраним дільником будуть відображатися тільки ті, які діляться або на 2, або на 3, або на 4, або на 5. Кнопка *Очистити* повертає форму до початкового вигляду, а *Завершити* — закриває вікно форми.

Початкове вікно Вікно після натиснення кнопки з цифрою 2

Завдання 3. Довідник (*12 балів*)

В обраному самостійно середовищі програмування розробіть проект *Довідник* із використанням вікон повідомлень, за допомогою якого користувач може отримати відомості про застосування комбінацій клавіш *Ctrl+C*, *Ctrl +V*, *Ctrl +X* при використанні буфера обміну в офісних програмах, наприклад, текстовому процесорі. Кількість форм вікон повідомлень, їх дизайн та об'єкти сплануйте самостійно.

РОЗДІЛ 6

АЛГОРИТМИ РОБОТИ З ОБ'ЄКТАМИ ТА ВЕЛИЧИНАМИ

20. ВЕЛИЧИНИ, ЇХ ТИПИ ТА ВЛАСТИВОСТІ

ПРИГАДАЙТЕ:

- як пов'язані інформація, повідомлення та дані;
- які існують типи даних;
- які пристрої використовує людина для роботи із даними.

ВИ ДІЗНАЄТЕСЯ:

- що таке величина та які властивості вона має;
- які розрізняють величини за типами;
- як описують змінні величини числового типу мовами програмування;
- як забезпечити введення даних користувачем при виконанні програми;
- як описати виконання операцій над числовими величинами мовою програмування;
- які стандартні функції можна використовувати в числових виразах.

Цікаво

Поняття величини прийшло до нас із математики. Уперше властивості величини чітко були сформульовані Евклідом у його «Початках» (III ст. до н.е.).

Цікаво

При написанні програм мовою *Free Pascal* в ідентифікаторах не розрізняють великі та малі літери. Наприклад, імена *a1* та *A1* — сприймаються програмою як однакові імена однієї й тієї ж величини. А в програмах, написаних мовою *Python*, навпаки, імена *a1* та *A1* відрізняються й можуть бути використані для позначення різних величин.

ВИВЧАЄМО

1. Що таке величина та які властивості вона має?

Для опису об'єктів і процесів у матеріальному світі ми використовуємо величини. З прикладами величин ви стикаєтеся щодня: відстань між будинком і школою, температура повітря тощо. За допомогою величин можна позначити довжину відрізка, площу земельної ділянки, висоту будинку, швидкість пішохода або автомобіля, час обертання планети навколо Сонця. Кожна величина характеризується певним значенням та одиницями, в яких вимірюється це значення, наприклад, швидкість може дорівнювати 80 км/год, відстань — 700 м, а температура — 15 °С. Поняття величини відіграє важливу роль у науці й відображає можливість фіксувати різні стани деяких об'єктів, зокрема, кількісну сторону проявів навколишньої дійсності. **Величина** має ім'я та може набувати різних значень із деякої множини допустимих значень. Тип цих значень визначає тип самої величини.

Визначення значень деяких величин можна здійснювати їх безпосереднім вимірюванням і за певним алгоритмом, якщо значення можуть змінюватися. Так, алгоритм розв'язування рівняння використовують для визначення значень невідомих величин — коренів рівняння. Комп'ютерна модель автомобіля, що подається за допомогою математичного рівняння, дає змогу визначати значення величини, що відповідає витратам палива, залежно від його швидкості. Із визначенням значень величин пов'язані також алгоритми одержання або створення текстів, різних списків тощо. У цих випадках величини набувають значень, що відповідають фрагментам тексту, елементам списків, значенням *Істина* або *Хибність* тощо.

Для посилань на величини у виразах під час створення програми використовують **імена** величин. Позначення імен називають також **ідентифікаторами**. Ідентифікатори добирають у вигляді деякого скінченного впорядкованого набору літер і цифр, який починається з літери або

символу підкреслення _. Прикладами ідентифікаторів величин можуть бути такі послідовності символів: *A*, *B2C*, *_I5*, *X*, *Y*, *SI*, *My_program*, *DAT_33* тощо. Хорошим стилем у програмуванні вважається, коли величинам надають імена, які б до деякої міри характеризували тип величини та її роль у програмі.

Величини поділяються на змінні та постійні (константи).

Величина, яка має одне й те саме значення в будь-які моменти часу, називається **постійною**, або **константою**.

Константам присвоюються значення в описовій частині програми і в процесі виконання програми їх змінювати заборонено. Для опису констант мовою програмування *Free Pascal* використовують службове слово `Const` (мал. 20.1). У мові програмування *Python* константи задаються у тексті програми.

```
Const
Max=1000;
Vxid='сегмент 5';
```

Мал. 20.1

Існують константи, до значень яких можна звертатися в програмі без попереднього опису (табл. 20.1).

Таблиця 20.1

Ідентифікатор	Значення	Опис
`True`	`True`	Істина
`False`	`False`	Хибність
`Maxint`	32767	Максимальне ціле

Цікаво

Службове слово *Const* — це скорочення від англ. *constant* — постійна.

Величина, яка в різні моменти часу може набувати різних значень, називається **змінною**.

Під час виконання програми в кожний момент часу величина, як правило, має деяке значення, яке називається **поточним** значенням. При цьому змінна величина може мати лише одне значення або не мати жодного. У процесі виконання програми величині може бути не надано ніякого конкретного значення. Тоді величина залишається **невизначеною**.

2. Які розрізняють величини за типами?

Кожна змінна та постійна величини належать до визначеного **типу**.

Числові величини — це величини, які можуть набувати значень з деяких числових множин. Наприклад, ціла числова величина *A* може набувати довільних значень із множини цілих чисел (..., –3, –2, –1, 0, 1, 2, 3,...).

Величини з текстовими значеннями можуть мати символьні або рядкові типи. **Символьні величини** можуть набувати значень із деякої множини символів, і кожне значення може містити лише один символ. **Рядкові величини** — це величини, що можуть набувати значень із деякої множини послідовностей символів, зокрема, слів або наборів слів. Наприклад, (*'понеділок'*, *'вівторок'*,..., *'неділя'*) — множина значень рядкової величини з іменем *День_тижня*.

Логічні величини можуть набувати тільки одного із двох значень: `True` (істина) або `False` (хибність). Від типу значень, яких може набувати

величина, залежить множина допустимих операцій. Наприклад, не можна виконувати арифметичні операції з текстовими величинами, операції ділення та віднімання над величинами логічного типу.

Тип величини — це сукупність множини допустимих значень і операцій, які дозволяється виконувати над цими значеннями.

Мал. 20.2

Тип величини визначає обсяг пам'яті, необхідний для зберігання її значень, а також структуру даних.

Тип величини характеризує як постійні, так і змінні величини (мал. 20.2).

3. Як описують величини числового типу мовами програмування?

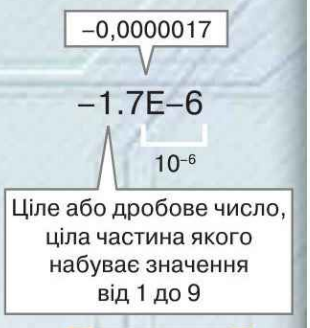
Для запису арифметичних виразів, аргументів математичних та деяких інших функцій можуть використовуватися числа або величини числового типу. Значення цих величин можуть бути цілими або дійсними числами.

Для опису числових величин мовою програмування *Free Pascal* використовують декілька службових слів. Це пов'язано з їх можливим діапазоном значень і відповідно обсягом, який вони можуть займати в пам'яті комп'ютера. Під час виконання програм, написаних мовою *Python*, система сама визначає обсяг, який числові величини можуть займати в пам'яті комп'ютера, залежно від введеного їх значення (табл. 20.2).

Таблиця 20.2

Мова програмування	Тип	Опис числових величин мовою програмування	Можливе значення
Free Pascal	Цілий	byte	0.. 255
		integer	-2147483648.. 2147483647
		smallint	-32768.. 32767
	Дійсний	real	Дійсне число
Python	Цілий	int	Довільне ціле число
	Дійсний	float	Дійсне число

Дійсні десяткові числа мовами програмування можна записати за звичайними правилами арифметики, проте в такому записі ціла частина від дробової відокремлюється десятковою крапкою, а не комою. Якщо десяткова крапка відсутня, число вважається цілим. Перед числом може записуватися знак «+» або «−». Якщо знак відсутній, за замовчуванням число вважається додатним. Наприклад, 9.23, 0.05. Для використання дуже великих чи дуже маленьких чисел їх подають у стандартному вигляді $a \cdot 10^n$ і записують у вигляді aEn, де позначення En замінює множення на 10 у степені n. Наприклад, число −0,0000017 можна записати у вигляді −1.7E−6 (мал. 20.3).



Мал. 20.3

Перед тим, як використовувати змінні величини у програмі, описують ім'я та тип кожної змінної.

У середовищі *Lazarus* кожна змінна має бути обов'язково описана у відповідному місці програми. Для цього після заголовка процедури перед тілом процедури, яке обмежується службовими словами `begin` та `end`,

розміщують розділ оголошення змінних, що розпочинається службовим словом `var`:

```
var ім'я_змінної: тип_змінної;
```

Якщо описується декілька змінних одного типу, то їх імена записують через кому.

```
var змінна_1, … ,змінна_N: тип_змінних;
```

Якщо програма міститиме змінні різних типів, то службове слово `var` записують один раз, а перелік змінних кожного іншого типу записують у новому рядку (мал. 20.4).

Мал. 20.4

Мова програмування *Python* є досить гнучкою й дає змогу створювати змінні будь-де в коді програми. Але гарний стиль оформлення програм передбачає, що змінні, за можливості, оголошуються в одному місці — їм надаються початкові значення. Це зручніше для програміста, який одразу бачить, які змінні будуть використовуватись і яких вони типів. Якщо конкретні значення не відомі або не потрібні на початку роботи, можна присвоїти спеціальне значення `None` — «нічого». Тобто це не нуль, не одиниця, не порожній рядок, а взагалі відсутність значення.

4. Як забезпечити введення даних користувачем при виконанні програми?

Щоб у процесі виконання програми користувач міг ввести деякі дані, які будуть опрацьовуватися, у програмі мовою *Python* можна використати функцію `input()`. Для цього необхідно записати команду присвоювання значення цієї функції змінній відповідного типу. Слід мати на увазі, що введена величина за замовчуванням буде рядкового типу. Щоб перетворити її в цілочисельний, використовують функцію `int`, у дійсний — `float`. Наприклад,

- `n=int(input('Введіть кількість:'))` — значення змінної n — ціле число, введене з клавіатури;
- `t=float(input('Введіть значення нормальної температури тіла людини:'))` — значення змінної t — дійсне число, введене з клавіатури;
- `a=input('Як тебе звати?')` — значення змінної a — рядкова величина, що введена з клавіатури.

У середовищі *PyCharm* у вікні виконання програми дані вводять після відповідного повідомлення.

У середовищі *Lazarus* для введення даних користувачем використовують функцію `InputBox()`, значення якої надають змінній відповідного типу. Результатом використання цієї функції, як і функції `input()` мовою *Python*, є значення рядкового типу. У результаті виконання команди присвоювання, що містить таку функцію, на екран буде виводитися **вікно введення**, що містить заголовок, текст підказки та поле введення, до якого з клавіатури слід ввести необхідне значення. Можна також передбачити значення за замовчуванням, яке буде надаватися змінній, якщо користувач не введе відповідне значення з клавіатури. Значення заголовка вікна,

Цікаво

Дані в комп'ютері зберігаються в комірках пам'яті, кожна з яких має свою адресу. Під час розробки комп'ютерних програм відразу не відомо, у яких комірках будуть зберігатись ті чи інші дані. Тому при написанні програм мовою програмування використовують змінні, що дає можливість програмісту під час складання програми звертатися не до адрес комірок пам'яті, а до ідентифікаторів.

Цікаво

Службове слово *var* — це скорочення від англ. *variable* — змінна.

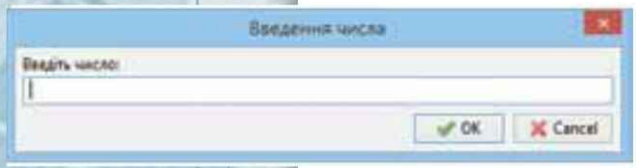
підказки для введення даних та значення за замовчуванням є агрументами функції `InputBox()`:

```
Змінна := InputBox('Заголовок', 'Підказка', 'Значення
за замовчуванням');
```

Якщо необхідно перетворити значення рядкового типу в числове або навпаки, використовують функції, подані в таблиці 20.3.

Таблиця 20.3

Функція	Призначення функції
`StrToInt` (*значення рядкового типу*)	Перетворення значення рядкового типу в значення цілочисельного типу
`StrToFloat` (*значення рядкового типу*)	Перетворення значення рядкового типу в значення дійсного типу
`FloatToStr` (*значення дійсного типу*)	Перетворення значення дійсного типу в значення рядкового типу
`IntToStr` (*значення цілого типу*)	Перетворення значення цілого типу в значення рядкового типу



Мал. 20.5

Наприклад, щоб змінна *n* при введенні даних користувачем із клавіатури набула цілого значення, у редакторі коду записують команду:

```
n := StrToInt(InputBox('Введення
числа', 'Введіть число:', ' '));
```

У результаті виконання такої команди буде відображено вікно введення (мал. 20.5). Якщо після введення даних у поле введення користувач обере кнопку *ОК*, то змінній *n* буде надано значення, введене користувачем. Якщо натисне *Cancel*, то змінна *n* набуде значення, що вказане як значення за замовчуванням.

ДІЄМО

Вправа 1. Туристична агенція.

Завдання. У середовищі програмування *Lazarus* розробіть для туристичної агенції проект реєстраційної форми на подорож містами Європи, у якій турист може вказати такі відомості:

- персональний код клієнта;
- орієнтовний місяць запланованої подорожі;
- кількість днів подорожі;
- кількість осіб, що подорожуватимуть разом із ним;
- тип харчування: 0,5 — сніданок або вечеря; 1,0 — триразове харчування; 1,5 — п'ятиразове харчування.

Після введення даних на екрані має з'являтися підтвердження введених даних. Формою реєстрації можна управляти за допомогою кнопок: *Почати реєстрацію*, яка викликає вікна введення даних, *Надіслати форму,* що закриває вікно головної форми.

1. Сплануйте проект. Передбачте, які об'єкти будуть використані на екранній формі та які події будуть відбуватися з ними.
2. У папці *Навчальні проекти* власної структури папок створіть папку *Форма_турагенція*.
3. Запустіть середовище *Lazarus*, створіть новий проект. Змініть властивості об'єкта `Form1` і розмістіть на формі об'єкти та надайте значень їх власти-

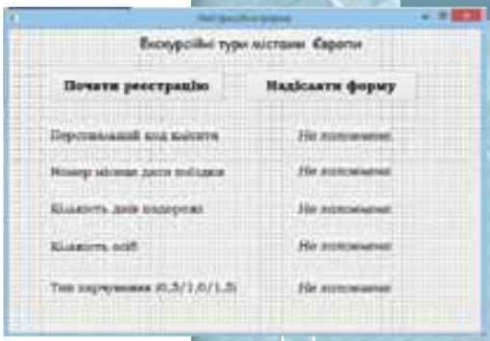
востей за зразком, поданим на малюнку 20.6.

4. Створіть процедуру опрацювання події: натиснуто кнопку *Почати реєстрацію*. У вікні редактора коду опишіть змінні, які будуть використовуватись у проекті:

```
kod, month, day, number: integer;
feeding: real;
```

5. У редакторі коду запишіть команди введення значень користувачем для всіх змінних величин. Наприклад, для змінної, яка набуде значення введеного персонального коду:

```
kod := StrToInt(InputBox('Введення даних',
'Введіть персональний код:', ''));
```



Мал. 20.6

6. Запишіть команди зміни значення властивості *Caption* усіх написів, для яких це значення *Не заповнене*. Наприклад, для напису, що відповідає персональному коду клієнта:

```
Label7.Caption:= IntToStr(kod);
```

7. Створіть процедуру опрацювання події: натиснуто кнопку *Надіслати форму*. У програмному коді процедури використайте метод *Закрити форму*.
8. Запустіть проект на виконання. Перевірте, чи дії, пов'язані з об'єктами управління екранної форми, відповідають умові завдання. Якщо є помилки — виправте їх.
9. Завершіть роботу з проектом і середовищем програмування.

5. Як описати виконання операцій над числовими величинами мовою програмування?

Ви вже знаєте, як у мовах програмування *Free Pascal* і *Python* використовується оператор присвоювання. Його також використовують і для надання значень змінним (табл. 20.4).

Таблиця 20.4

Мова програмування	Оператор присвоювання	Команда присвоювання	Приклад
Free Pascal	:=	*<ім'я змінної>:=<вираз>*	*a*:=5
Python	=	*<ім'я змінної>=<вираз>*	*a*=5

У програмі мовою *Python* надавати значення кільком змінним можна однією командою (мал. 20.7).

$a, b, c = 7.5, 2.3, 8$

імена змінних — список значень

Мал. 20.7

Вираз, розташований праворуч від символу оператора присвоювання, задає порядок виконання дій над даними і складається з операндів (констант, змінних, звертань до функцій), круглих дужок та знаків арифметичних операцій, наприклад, $a+b*sin(cos(x))$. Результат обчислення виразу надається змінній, ім'я якої записане ліворуч. Складність виразу не обмежується. Значення виразу має бути узгодженим із типом змінної, якій воно присвоюється. Наприклад, змінній дійсного типу можна присвоїти значення дійсного або цілого типу, а змінній цілого — тільки значення цілого типу.

Над числовими величинами можна виконувати:

1) арифметичні операції додавання (+), віднімання (−), множення (*), ділення (/);

2) операції цілочисельної арифметики (застосовуються тільки до даних цілого типу) (табл. 20.5).

Таблиця 20.5

Операція		Призначення	Приклад	
Free Pascal	*Python*		*Free Pascal*	*Python*
`div`	//	ділення націло (повертає цілу частину дробу, дробова частина відкидається)	17 div 10 = 1	17//10 = 1
`mod`	%	остача від ділення	17 mod 10 = 7	17%10 = 7

У програмі мовою *Python* є деякі особливості використання операцій додавання та віднімання:

x+=3 — збільшення значення змінної x на 3;

x−=2 — зменшення значення змінної x на 2.

ДІЄМО

Вправа 2. Сума цифр.

Завдання. У середовищі програмування *Lazarus* розробіть проект *Суматор*, у якому після натиснення кнопки *Старт* у вікні введення даних слід ввести трицифрове ціле число. Після чого у вікно повідомлення буде виводитися сума його цифр.

Таблиця 20.6

Ідентифікатор	Тип	Призначення
`chislo`	`integer`	Введене трицифрове число
`dig1`	`integer`	Перша цифра числа
`dig2`	`integer`	Друга цифра числа
`dig3`	`integer`	Друга цифра числа
`s`	`integer`	Сума цифр

1. Сплануйте проект. Передбачте, які об'єкти будуть використані на екранній формі та які події будуть відбуватися з ними.

2. У папці *Навчальні проекти* власної структури папок створіть папку *Суматор*. Виконайте інші дії підготовчого етапу розробки проекту.

3. У вікні дизайнера форми середовища *Lazarus* додайте об'єкт — кнопка, надайте значення її властивості *Caption*: *Старт.*

4. Створіть процедуру опрацювання події: *натиснуто кнопку Старт*. Опишіть змінні величини, які будуть використовуватися (табл. 20.6):

5. Запишіть команду введення значення для змінної *chislo*:

```
chislo:= StrToInt(InputBox('Введення числа,
'Введіть трицифрове число:', ''));
```

6. Проаналізуйте та введіть програмний код виділення цифр числа:

```
dig1:=chislo div 100;
chislo:= chislo-dig1*100;
dig2:=chislo div 10;
dig3:= chislo-dig2*10;
s:= dig1+ dig2+ dig3;
```

7. Додайте до програмного коду команду створення вікна повідомлення:

```
MessageDlg(IntToStr(s), mtInformation,[mbOk], 0);
```

8. Запустіть проект на виконання. Перевірте для введеного числа 732, ви маєте отримати результат 12.

9. Завершіть роботу з проектом і середовищем програмування.

6. Які стандартні функції можна використовувати в числових виразах?

У виразах, які записують у середовищі програмування для виконання обчислень, можна використати стандартні функції, деякі з них ви вже використовували на уроках математики або вивчатимете пізніше (табл. 20.7).

Таблиця 20.7

Функція	Тип аргументу	Тип результату	Результат
`abs(x)`	Цілий/дійсний	Цілий/дійсний	Модуль числа
`sin(x)`	Дійсний	Дійсний	Синус числа
`cos(x)`	Дійсний	Дійсний	Косинус числа
`pi`	Без аргумента	Дійсний	Значення числа π
`sqr(x)`	Дійсний	Дійсний	Квадрат числа
`sqrt(x)`	Дійсний	Дійсний	Квадратний корінь
`int(x)`	Дійсний	Дійсний	Ціла частина числа
`frac(x)`	Дійсний	Дійсний	Дробова частина числа
`round(x)`	Дійсний	Цілий	Округлення числа
`trunc(x)`	Дійсний	Цілий	Відкидання дробової частини числа
`random(n)`	Цілий	Цілий	Випадкове число від 0 до *n*

Щоб використовувати вказані функції у програмі мовою *Python* у середовищі *PyCharm*, у програмному коді звертаються до бібліотеки `math`:

```
from math import cos, pi
y=cos(pi)
```

ДІЄМО

Вправа 3. Доріжки в парку.

Завдання. У деякому місті всі пішохідні доріжки побудували на перпендикулярних лініях. Але це не був найкоротший шлях. Розробіть проект у середовищі *PyCharm*, за допомогою якого можна буде представити мерії деякого міста розрахунок довжини нових коротких доріжок.

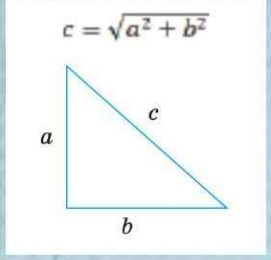
1. Сплануйте проект. Визначте вхідні дані та результат. Побудуйте інформаційну модель завдання. Візьміть до уваги, що нею може бути графічна модель (мал. 20.8).
2. Запустіть середовище програмування *PyCharm*.
3. Створіть новий файл програми мовою *Python* з іменем *Доріжки* в папці *Навчальні проекти* власної структури папок.
4. У вікні редактора коду введіть команду приєднання бібліотеки математичних функцій:

```
from math import sqrt
```

5. Запишіть у вікні редактора коду команди для введення значень для змінних a, b:

```
a=int(input('a= '))
b=int(input('b= '))
```

6. Запишіть вираз для знаходження значення змінної та команду для виведення результату на екран за правилами мови *Python*:



Мал. 20.8

```
c = sqrt(a*a+b*b)
print(c)
```

7. Запустіть проект на виконання. Перевірте, чи результат для значень змінних $a = 3$, $b = 4$ становить *5.0*.
8. Завершіть роботу з проектом і середовищем.

ОБГОВОРЮЄМО

Обговоріть запитання, що містяться у файлі *Тема 20*, який зберігається у папці *Обговорюємо*. Також відкрити ці запитання можна за допомогою QR-коду.

ПРАЦЮЄМО В ПАРАХ

1. Запропонуйте одне одному приклади числових величин, значення яких міститься в діапазонах, описаних у таблиці 20.2. Один називає опис типу величини мовою програмування, інший — приклад.
2. Пограйте у гру «Значення величини — тип величини». Один учасник називає множину значень деякої величини, а інший — відповідний тип величини. Гра продовжується до першої помилки, після цього учасники міняються ролями.
3. Запропонуйте одне одному вирази для обчислення значення змінної-результату, у яких вхідними даними є цілі числа, а результатом — дійсне число. Запишіть програмний код оголошення змінних та обчислення значення змінної-результату: один з учасників — мовою програмування *Free Pascal*, інший — мовою *Python*. Обміняйтеся записами та перевірте одне одного.
4. Обговоріть, як змінити проект *Доріжки в парку*, створений у середовищі *PyCharm*, щоб реалізувати його в середовищі *Lazarus*. Результати обговорення використайте при створенні проекту.
5. Обговоріть, як змінити проект *Суматор*, створений у середовищі *Lazarus*, щоб реалізувати його в середовищі *PyCharm*. Результати обговорення використайте при створенні проекту.

ПРАЦЮЄМО САМОСТІЙНО

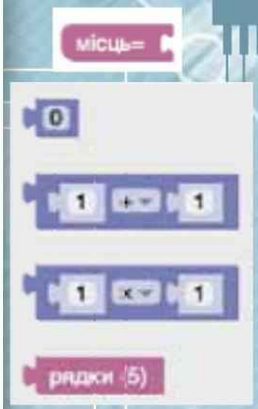
1. Запишіть вираз для обчислення кількості місць *K* в літаку при зміні рядів мовами програмування *Free Pascal* й *Python,* якщо:
   а) у літаку кілька рядів пасажирських сидінь. Кожний ряд *R* містить чотири місця;
   б) у літаку два місця в кабіні екіпажу (пілот і другий пілот) і кілька рядів пасажирських сидінь. Кожний ряд *R* містить чотири місця;
   в) у літаку два місця в кабіні екіпажу (пілот і другий пілот), кілька рядів R_1 1-го класу та кілька рядів R_2 2-го класу пасажирських місць. Кожний ряд 1-го класу містить чотири місця. Кожний ряд 2-го класу містить п'ять місць.

   Створіть відповідні вирази в навчальному онлайновому середовищі *Blokly* за адресою: *https://blockly-demo.appspot.com/static/demos/plane/index.html?lang=uk/*

Використайте блоки для запису арифметичних операцій додавання та множення, запису кількості рядів і числових значень (мал. 20.9).

Змінюючи положення повзунка для зміни кількості рядів, визначте кількість місць у літаку, якщо літак міститиме 8, 12, 17 рядів для завдань *а*) та *б*) (мал. 20.10), та 3, 4 або 7 рядів першого класу —для завдання *в*).

Мал. 20.10



Мал. 20.9

2. Розробіть проект *Сімейний депозит*, за допомогою якого можна визначити, яку суму отримає родина в кінці року, якщо на його початку відкриє в банку депозитний рахунок у розмірі *S* тис. грн під 18 % річних. Середовище програмування оберіть самостійно. Перевірте виконання проекту для $S = 10\,000$.

3. У середовищах програмування *Lazarus* та *PyCharm* розробіть проект *Електрика*, за допомогою якого можна визначити опір електричного кола *R*, якщо в ньому провідники з опорами *R1, R2, R3, R4* з'єднані:

1) послідовно $R = R1 + R2 + R3 + R4$;

2) паралельно $R = \frac{1}{R_1} + \frac{1}{R_2} + \frac{1}{R_3} + \frac{1}{R_4}$.

Порівняйте програмні коди знаходження значення змінної *R*. Перевірте виконання проекту для $R1 = 2$, $R2 = 3$, $R3 = 4$, $R4 = 5$ Ом.

4. Для швейної компанії розробіть проект із графічним інтерфейсом *Залишок*, за допомогою якого користувач планового відділу буде визначати площу тканини, яка залишиться після того, як із заготовки у формі квадрата зі стороною *a* вирізали круг радіусом *R*. Перевірте виконання проекту для $a = 4$ м, $R = 2$ м.

5. Дано змінні *x, y* дійсного типу. Розробіть проект для обчислення значення змінної *z* для $x = 1$, $y = -5$:

1) $z = \frac{|x|+|y|}{1+|xy|}$;

2) $z = 1 + |y - x| + \frac{(y-x)^2}{2} + \frac{|y-x|^3}{3}$;

3) $z = \frac{2y}{\cos(x - \pi/6)}$;

4) $z = \frac{x}{1+\frac{x^2}{2+(2x)^2}}$.

6. Для ведення домашнього бізнесу сплануйте та розробіть проект *Домашня бухгалтерія*, у якому можна ввести доходи всіх членів сім'ї за місяць й обов'язкові виплати з бюджету на комунальні платежі, транспорт, зв'язок, Інтернет, інші обов'язкові витрати та розрахувати залишок сімейного бюджету на наступний місяць. Дизайн та елементи управління доберіть самостійно. Для перевірки виконання проекту введіть дані про бюджет власної родини.

21. ПРАКТИЧНА РОБОТА 10

СКЛАДАННЯ ТА ВИКОНАННЯ ЛІНІЙНИХ АЛГОРИТМІВ ОПРАЦЮВАННЯ ВЕЛИЧИН У СЕРЕДОВИЩІ ПРОГРАМУВАННЯ

ПРИГАДАЙТЕ

- як описують змінні величини мовою програмування;
- як вводити значення величин у програмі;
- які операції можна виконувати над числовими величинами.

СТВОРІТЬ

У власній структурі папок створіть папку *Практична робота 10*.

ПАМ'ЯТАЙТЕ

Під час виконання практичних завдань пам'ятайте про правила безпеки життєдіяльності при роботі з комп'ютером!

Завдання1. Парк атракціонів (*11 балів*)

У парку атракціонів побудували нову гірку завдовжки c, яка біля підніжжя нахилена до основи b під кутом α градусів. Розробіть проект *Атракціон*, за допомогою якого можна визначити висоту гірки a та довжину основи b, округлені до цілого числа. Використайте формулу для переведення величини, вираженої у градусах, у радіани:

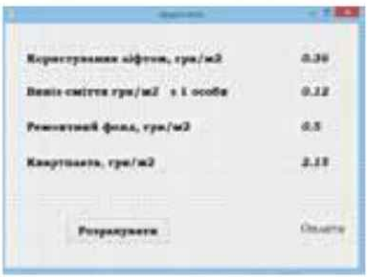
$$\text{радіани} = \text{градуси} \cdot \alpha/180$$

Завдання 2. Тривалість рейсу (*11 балів*)

Розробіть проект *Тривалість рейсу* для обчислення тривалості рейсу в годинах і хвилинах для деякого автотранспортного підприємства, якщо відомо, що маршрут складається із трьох відрізків — $s1$, $s2$, $s3$ км, які транспорт долає зі швидкістю $v1$, $v2$, $v3$ км/год. Між ділянками маршруту є зупинки по t хв.

Завдання 3. Квартплата (*18 балів*)

В об'єднанні співвласників багатоквартирного будинку здійснюються такі виплати: 0,36 грн/м^2 — за користуванням ліфтом, 0,12 грн/м^2 — за вивіз сміття з особи, 0,5 грн/м^2 — формування ремонтного фонду, 2,15 грн/м^2 — квартплата.



У середовищі програмування *Lazarus* розробіть проект *Квартплата* для розрахунку оплати відповідно до показників квитанції за квартиру площею S м^2 на місяць, у якій проживає n осіб. Передбачте введення значень площі та кількості осіб у вікнах введення, а відображення розміру оплати — на формі проекту.

Завдання 4. Перетворення тексту (*10 балів*)

Для перетворення тексту у середовищі програмування *PyCharm* розроблено проект *Кодування*. Реалізуйте проект для тестових значень, обраних відповідно до вказівок програми. Встановіть, який алгоритм перетворення тексту використано у програмі.

22. ТЕКСТОВІ ВЕЛИЧИНИ ТА ОПЕРАЦІЇ НАД НИМИ

ПРИГАДАЙТЕ:

- які розрізняють об'єкти текстового документа;
- які дії виконують під час редагування текста;
- як опрацьовують документи в текстових процесорах.

ВИ ДІЗНАЄТЕСЯ:

- як вводять та описують текстові величини мовами програмування;
- які операції виконують над текстовими величинами;
- які функції застосовують для опрацювання текстових величин;
- якими бувають помилки при створенні та виконанні програм;
- як налагоджують програми в середовищах програмування.

ВИВЧАЄМО

1. Як вводять та описують текстові величини мовами програмування?

Для опрацювання текстових даних мовою програмування використовують величини символьного та рядкового типів.

Значенням величини **символьного типу** може бути один символ — літера, цифра чи знак. Набір символів утворює **рядок**.

Надавати значення текстовим величинам можна різними способами:

- при написанні програмного коду за допомогою оператора присвоювання;
- під час виконання програми, що містить команди або вікна введення;
- за допомогою елемента управління *текстове поле*.

Текстове поле створюється на екранній формі проекту в середовищі *Lazarus* за допомогою компонента *Edit* (*Поле введення*) (мал. 22.1).

Мал. 22.1

Текст, який записаний у полі введення, визначається значенням властивості `Text` (*Текст*), і його можна редагувати після запуску проекту на виконання.

Окрім властивостей, які мають інші елементи управління, наприклад `Label` (*Напис*), для текстового поля можна визначити додаткові (табл. 22.1).

Таблиця 22.1

Властивість	Опис
`ParentFont`	Наслідування компонентом значень параметрів шрифта форми, на якій розташований компонент. Якщо ця властивість має значення `True`, то при зміні властивостей `Font` форми автоматично змінюється значення властивості `Font` компонента *текстове поле*
`Enabled`	Обмеження можливості змінити текст у текстовому полі. Якщо ця властивість має значення *False*, то текст у текстовому полі редагувати не можна

У програмах, створених мовою програмування *Python*, після підключення модуля віконного графічного інтерфейсу користувача також можна використати текстове поле для введення даних. Для цього призначена функція `tkinter.Entry` (мал. 22.2).

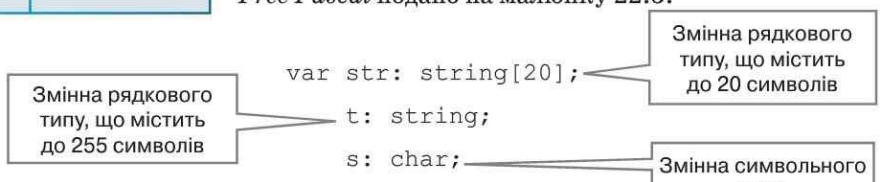
Мал. 22.2

Текстові величини, як і числові, описують у програмі в розділі опису змінних, вказуючи відповідний тип. У мовах програмування для опису текстових величин використовують службові слова (табл. 22.2).

Таблиця 22.2

Тип величини	Опис мовою програмування	
	Free Pascal	*Python*
Символьний	*char*	*str*
Рядковий	*string*	

Наприклад, опис змінних мовою програмування *Free Pascal* подано на малюнку 22.3.



Мал. 22.3

У програмному коді мовою програмування *Free Pascal* текстове значення записують між символами одинарних лапок (‘). Наприклад:

```
s:='Q'; s1:='data';
```

У програмному коді мовою програмування *Python* можна використовувати одинарні та подвійні лапки. Якщо для позначення рядка застосовано подвійні лапки, всередині може вільно використовуватись апостроф, а подвійні лапки інтерпретатор вважатиме кінцем рядка. І навпаки, якщо рядок позначено одинарними лапками, всередині можуть міститися подвійні. Крім того, для позначення рядків можуть використовуватися три пари одинарних або подвійних лапок. У такому разі все, що міститься між ними, відображатиметься так, як записано. Такий спосіб надання значень текстовим величинам зручно використовувати для подальшого структурованого виведення даних (мал. 22.4).

```
print(''' День
перший''')
```

```
День
перший
```

Мал. 22.4

Цікаво

Поняття **escape-послідовності** походить від англомовного аналогу *escape sequences* (від англ. *Escape* — уникати), використовуються для запису в текстових значеннях спеціальних символів, у яких уникається виведення на екран самої послідовності символів.

Рядки можуть містити набори символів, що починаються з похилої риски \ — вони називаються **escape-послідовностями**. При виведенні рядка такі символи опрацьовуються спеціальним чином і можуть повністю або частково не відображатися на екрані. Найбільш часто використовуються такі:

- \n — перехід на новий рядок;
- \t — вставлення табуляції;
- \" — подвійні лапки (корисно, якщо необхідно вставити подвійні лапки в тексті українською мовою, взятому в одинарні лапки);
- \' — одинарні лапки (корисно, якщо необхідно вставити, наприклад, апостроф у рядок, взятий в одинарні лапки).

Таблиця 22.3

Мова програмування	Змінна	Символ
Free Pascal	`z:='школа'`	`z[4] – л`
Python	`z='школа'`	`z[4] – а`

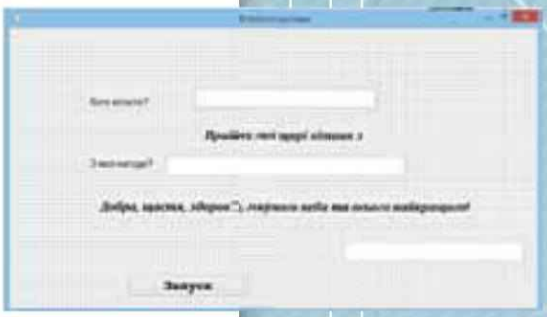
Усі символи в рядку мають свій порядковий номер. За цим номером можна отримати значення сим-

волу, вказавши номер у квадратних дужках після імені змінної. У мові програмування *Free Pascal* нумерація символів розпочинається з одиниці, а в *Python* — з нуля (табл. 22.3).

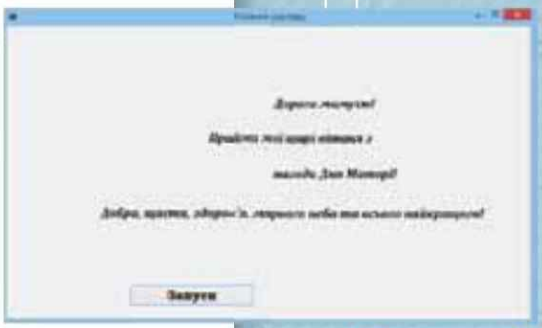
ДІЄМО

Вправа 1. Вітальна листівка.

Завдання. У середовищі програмування *Lazarus* розробіть проект *Вітальна листівка* за зразком (мал. 22.5), у якому у відповідні текстові поля на екранній формі користувач вносить ім'я адресата привітання, подію та ім'я автора привітання.



Мал. 22.5



Мал. 22.6

1. Сплануйте проект. Передбачте, які об'єкти будуть використані на екранній формі та які події відбуватимуться з ними.
2. У середовищі програмування *Lazarus* розробіть проект *Вітальна листівка* за зразком.
3. Запустіть середовище *Lazarus*, створіть новий проект. Змініть властивості об'єкта `Form1`, розмістіть на формі об'єкти та надайте значень їх властивостям, щоб після запуску проекту на виконання можна було отримати привітання, наприклад, подане на малюнку 22.6.
4. Створіть процедуру опрацювання події: натиснуто кнопку *Сформувати*. У вікні редактора коду опишіть змінні, які будуть використовуватись у проекті: `person` (адресат), `event` (подія), `nik` (ім'я автора). Укажіть їх тип — `string` (рядковий).
   Проаналізуйте фрагмент програмного коду (мал. 22.7).

```
person:=edit1.Text;
label5.Caption:=person;
label5.visible:=true;
edit1.visible:=false;
label1.visible:=false;
```

Змінній *person* присвоїти значення, яке вводиться в текстовому полі

Вміст текстового поля перенести в напис, який стає видимим

Значення текстової змінної перенести в напис, який стає видимим

«Зайві» елементи управління зробити невидимими

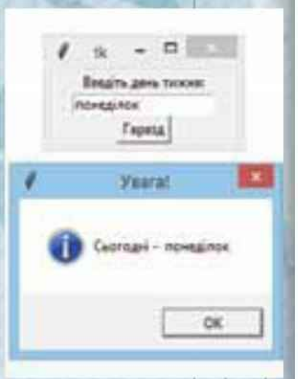
Мал. 22.7

5. У вікні редактора коду запишіть команди, за допомогою яких текст, введений у текстові поля екранної форми, присвоєно значенням властивості `Caption` написів 5–7 на формі. При цьому текстові поля та написи із запитаннями стануть невидимими.
6. Запустіть проект на виконання. Перевірте, чи дії, пов'язані з об'єктами управління екранної форми, відповідають умові завдання. Якщо є помилки — виправте їх.
7. Завершіть роботу з проектом і середовищем програмування.

Вправа 2. Нагадування.
Завдання. У середовищі програмування *PyCharm* розробіть проект, у якому користувач буде вводити день тижня, натискати кнопку *Гаразд*, й отримувати у вікні із заголовком *Увага*! таке повідомлення: *Сьогодні — <день тижня, який було введено>*.

1. Запустіть середовище програмування *PyCharm*.
2. Створіть новий файл програми мовою *Python* з іменем *Нагадування* в папці *Навчальні проекти* власної структури папок.
3. У вікні редактора коду запишіть команди, які зберігаються у файлі *Код_Нагадування* в папці *Програмування*.
4. Змініть значення властивостей об'єктів, які використовуються у програмному коді, так, щоб вони реалізували завдання.

```
import tkinter
import tkinter.messagebox
main = tkinter.Tk()
# створення об'єкта для отримання значення з текстового поля
str_var = tkinter.StringVar()
# опрацювання події натиснення кнопки
def button_click():
    tkinter.messagebox.showinfo("Program", "Hello" + str_var.get())
# створення текстового напису та його розміщення на головній формі
label = tkinter.Label(text="Enter Your Name:")
label.pack()
# створення текстового поля та його розміщення на головній формі
edit = tkinter.Entry(main, textvariable = str_var)
edit.pack()
# створення кнопки та розміщення об'єкта на головній формі
button = tkinter.Button(main, text="Push Me!",
command=button_click)
button.pack()
# запуск опрацюваня подій програми
main.mainloop()
```



Мал. 22.8

5. Запустіть проект на виконання. Перевірте, чи результат для введеного дня тижня *понеділок* відповідає малюнку 22.8.
6. Завершіть роботу з проектом і середовищем.

Таблиця 22.4

Мова програмування	Приклад набору команд	Результат — значення змінної *R*
Free Pascal	`F:='алго'; L:='ритм'; R:=F+L;`	`'алгоритм'`
Python	`F='алго' L='ритм' R=F+L;`	`'алгоритм'`

2. Які операції виконують над текстовими величинами?

Над текстовими величинами виконують операцію склеювання — поєднання кількох рядкових величин, яка позначається символом «+» (табл. 22.4).

Текстові величини є незмінюваними, це означає, що не можна змінити частину рядка, не створивши нового.

У програмі мовою програмування *Python* при роботі з текстовими величинами можна використовувати операцію зрізу, за допомогою якої копіюється послідовність або її частина. Наприклад, для змінної

`s = 'Hello world'` операції зрізу подані в таблиці 22.5.

Таблиця 22.5

Опис	Приклад команди	Результат — значення змінної *s1*
Зріз від початку рядка до символу з номером *n*	`s1 = s[:6] + 'Python!'`	`'Hello Python!'`
Зріз від символу з номером *n* рядка включно до символу з номером *m*	`s1=s[3:5]`	`'lo'`
Зріз від символу з номером *n* рядка включно до кінця рядка	`s1=s[6:]`	`'world'`

Щоб записати символи в рядку у зворотному порядку, використовують операцію

```
s3=s[::-1].
```

Тоді змінна `s3` матиме значення `'dlrow olleH'`.

ДІЄМО

Вправа 3. Слова.

Завдання. Складіть програму мовою *Python*, за допомогою якої із введеного слова *інформатика* будуть утворюватися слова *форма*, *романтика*.

1. Запустіть середовище програмування *PyCharm*.
2. Створіть новий файл програми мовою *Python* з іменем *Слова* у папці *Навчальні проекти* власної структури папок.
3. Нехай у програмі будуть використані змінні: *s* — значенням якої буде введене з клавіатури слово *інформатика*, *s1* — що має отримати значення форма, *s2* — що має отримати значення *романтика*.
4. Визначте номер позиції кожної літери у слові.

0	1	2	3	4	5	6	7	8	9	10
і	н	ф	о	р	м	а	т	и	к	а

5. У вікні редактора коду запишіть команди введення змінної *s* та знаходження значень змінних *s1*, *s2*:

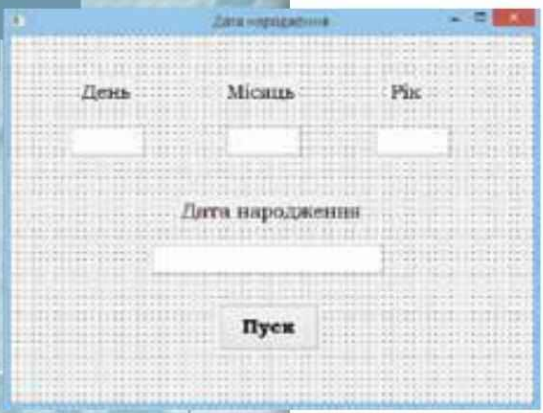
```
s=input('Введіть слово')
s1=s[2:7]
s2=s[4]+s[3]+s[5:7]+s[1]+s[7:]
```

6. Запишіть команди виведення отриманих значень.
7. Запустіть проект на виконання. Введіть у вікні виконання програми значення змінної *s*: *інформатика*. Перевірте отримані результати. За потреби виправте помилки.
8. Завершіть роботу із середовищем.

Вправа 4. Дата народження.

Завдання. Розробіть проект у середовищі *Lazarus*, у якому в текстові поля, розміщені у верхній частині форми, користувач вводить день, місяць і рік свого народження та після натиснення кнопки *Пуск* у відповідному текстовому написі формується дата народження.

1. Сплануйте проект. Передбачте, які об'єкти будуть використані на екранній формі та які події будуть відбуватись із ними.
2. У папці *Навчальні проекти* власної структури папок створіть папку *Дата_народження*.



Мал. 22.9

3. Запустіть середовище *Lazarus*, створіть новий проект. Змініть властивість `Caption` об'єкта `Form1`, надавши їй значення *Дата народження*. Розмістіть на формі об'єкти за зразком і надайте значень їх властивостей, визначених самостійно. Властивості `Enabled` текстового поля для виведення результату надайте значення `False` (мал. 22.9).
4. Створіть процедуру опрацювання події: натиснуто кнопку *Пуск*. У вікні редактора коду опишіть змінні, які будуть використовуватись у проекті:

```
var d, m, y, rez: string;
```

5. У вікні редактора коду запишіть команди присвоєння значень змінним *d*, *m*, *y* за зразком:

```
d:=edit1.text;
```

6. Запишіть вираз для отримання значення змінної *rez*:

```
rez:= d + '.' + m + '.' + y;
```

7. Надайте значення отриманої змінної властивості *Text* текстового поля для відображення повної дати народження.
8. Запустіть проект на виконання. Введіть дані в текстові поля, місяць народження запишіть числом. Натисніть кнопку *Пуск* і перевірте правильність роботи програми. За потерби виправте помилки.
9. Завершіть роботу з проектом і середовищем.

3. Які функції застосовують для опрацювання текстових величин?

Ви вже використовували функції для перетворення текстової величини у величину числового типу та навпаки.

У програмах мовою програмування *Free Pascal* використовують й інші функції для опрацювання текстових величин (табл. 22.6).

Таблиця 22.6

Опис функції мовою програмування	Тип аргументу	Тип результату	Призначення
`length (S)`	*S* — текстовий	Цілий (byte)	Визначення кількості символів у рядку *S*
`copy(S,n,m)`	S — текстовий *n*, *m* — цілий	Текстовий	Копіювання *n* символів рядка *S*, починаючи з позиції *m*
`delete(S,n,m)`	*S* — текстовий *n*, *m* — цілий	Текстовий	Видалення *n* символів рядка *S*, починаючи з позиції *m*
`insert(S,S1, m)`	*S*, *S1* — текстовий, *m* — цілий	Текстовий	Вставлення рядка *S1* у рядок *S*, починаючи з позиції *m*
`pos(S1,S2)`	*S1*,*S2* — текстовий	Цілий	Номер позиції, з якої рядок *S2* входить у рядок *S1*

При роботі з текстовими величинами, які містять символи українського алфавіту, ці функції працюють некоректно. Щоб цього уникнути, в розділ *uses* додають модуль *LCLProc*:

```
uses LCLProc;
```

Тоді перед іменем кожної функції, що подано в таблиці 22.6, додають *UTF8*. Наприклад, при використанні латинських літер використовується функція `length('s')`, а при використанні символів кирилиці — `UTF8length('ф')`.

У мові програмування *Python* для опрацювання текстових величин використовують функції, наведені в таблиці 22.7.

Таблиця 22.7

Опис функції мовою програмування	Призначення
`len(S)`	Визначення кількості символів у рядку *S*
`s.upper()`	Змінює регістр усіх символів рядка на верхній
`s.lower()`	Змінює регістр усіх символів рядка на нижній
`s.replace(s_old, s_new)`	Замінює всі входження фрагмента *s_old* у рядку на *s_new*
`s.replace(s_old, s_new, count)`	Замінює перші *count* входжень фрагмента *s_old* у рядку на *s_new*
`s.find(s1)`	Повертає позицію входження (індекс першого символу) фрагмента *s1* у рядку, або –1, якщо фрагмент не знайдено
`s.find(s1,start_pos)`	Повертає позицію входження (індекс першого символу) фрагмента *s1* у рядку, починаючи з позиції *start_pos*, або –1, якщо фрагмент не знайдено

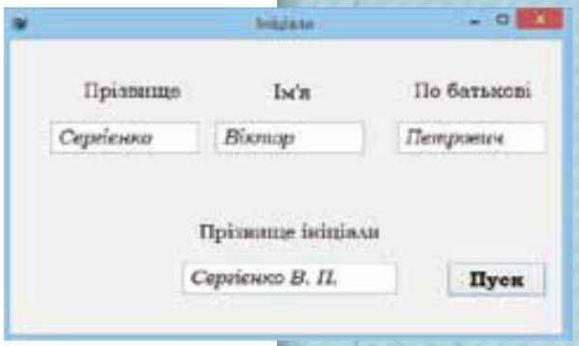
У мовах програмування *Free Pascal* і *Python* є однакові функції для роботи з текстовими величинами (табл. 22.8).

Таблиця 22.8

Опис функції мовою програмування	Тип аргументу	Тип результату	Призначення
`chr(x)`	Цілий	Символьний	Визначає символ із кодом *x*
`ord(c)`	Символьний	Цілий	Визначає код символа *c*

Вправа 5. Ініціали.

Завдання. Розробіть у середовищі *Lazarus* проект, після запуску якого до текстових полів екранної форми користувач вводитиме прізвище, ім'я і по батькові. Після натиснення кнопки *Пуск* до відповідного текстового поля будуть виводитися прізвище та ініціали (мал. 22.10).



Мал. 22.10

1. Сплануйте проект. Передбачте, які об'єкти будуть використані на екранній формі та які події будуть відбуватись із ними.
2. У папці *Навчальні проекти* власної структури папок створіть папку *Ініціали*.
3. Запустіть середовище *Lazarus*, створіть новий проект. Змініть властивість `Caption` об'єкта `Form1`, надавши їй значення *Ініціали*. Розмістіть на формі об'єкти за зразком і надайте значень їх властивостей, визначених самостійно. Властивості *Enabled* текстового поля для відображення прізвища та ініціалів надайте значення `False`, щоб користувач не міг змінити вміст цього поля з клавіатури.

4. Створіть процедуру опрацювання події: натиснуто кнопку *Пуск*. У вікні редактора коду опишіть змінні, які будуть використовуватись у проекті:
```
var last_name, first_name, sname, full_name: string;
```
5. У вікні редактора коду запишіть команди присвоювання значень змінним *last_name, first_name, sname*.
6. Запишіть команду знаходження значення змінної величини *full_name:*
```
full_name:= last_name+' ' +UTF8copy(first_name, 1, 1)+
UTF8copy(sname, 1, 1);
```
7. Передбачте виведення отриманого значення змінної у відповідне текстове поле.
8. Запустіть проект на виконання. Введіть дані в текстові поля. Натисніть кнопку *Пуск* і перевірте правильність роботи програми. За потреби виправте помилки.
9. Завершіть роботу з проектом і середовищем.

4. Якими бувають помилки при створенні та виконанні програм?

Створений у середовищі програмування проект може не виконуватися взагалі, або результат його виконання не відповідатиме очікуваному результату. Це відбувається тоді, коли при складанні алгоритму розв'язування завдання чи написанні коду програми були допущені помилки. Розрізняють три групи помилок:

- синтаксичні;
- помилки під час виконання;
- логічні.

Синтаксичні помилки можна виявити як у процесі написання програмного коду, так і після запуску проекту на виконання. Якщо деяка команда у програмному коді написана програмістом не за правилами, прийнятими в мові програмування, то вона може набувати іншого кольору, відмінного від кольору правильно написаних команд (мал. 22.11).

Команди записані правильно

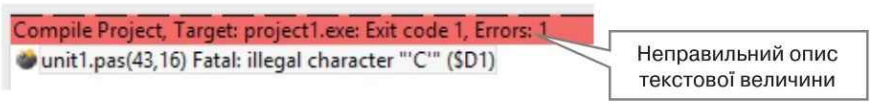
```
Edit3.Text:=Edit1.Text;

Edit4.Text:= текст;
```

У команді значення текстової змінної не взято в апостроф

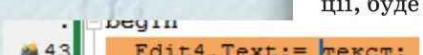
Мал. 22.11

Після запуску проекту на виконання, якщо у програмному коді така помилка не була виправлена, у середовищі *Lazarus* у вікні повідомлення про хід компіляції проекту отримуємо номер рядка програмного коду та позиції об'єкта в рядку, де допущена помилка, й опис помилки (мал. 22.12).



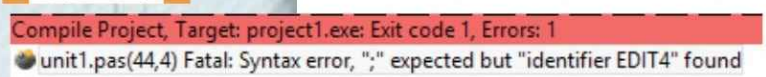
Мал. 22.12

У вікні редактора коду рядок, на якому «зупинився» процес компіляції, буде позначено (мал. 22.13).



Мал. 22.13

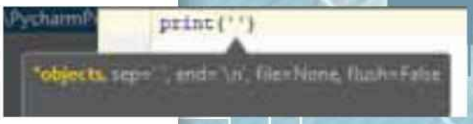
Рядок, у якому допущена помилка, може бути вказаний не точно. Наприклад, повідомлення про пропущений символ «;» у рядку під номером 43 у вікні повідомлення буде позначено рядком із номером 44 (мал. 22.14).



Мал. 22.14

У середовищі *PyCharm* при введенні команди «підсвічується»

вікно повідомлення з підказкою щодо правильного синтаксису (мал. 22.15).



Мал. 22.15

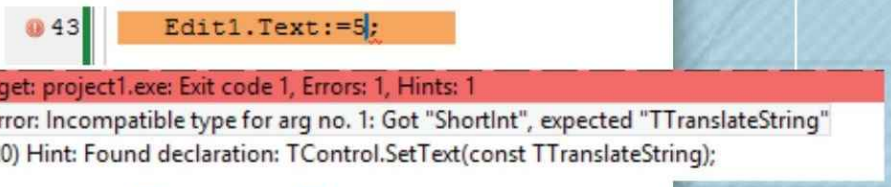
Але якщо програміст все-таки припустився помилки, то в коді проекту команда, вжита не за правилами синтаксису мови, буде виділена червоною хвилястою лінією `print('слово`. Після запуску проекту на виконання у вікні виконання програми середовища буде відображено не тільки місце розташування помилки у програмному коді, а й сама команда, у якій допущена помилка (мал. 22.16).

```
  File "D:/підручник/8 клас/Розділ 5/pomilky.py", line 1
    print('слово)
                ^
SyntaxError: EOL while scanning string literal
```

Відсутній символ завершення текстової величини

Мал. 22.16

Типовими є помилки, що позначають невідповідність типів описаних величин і значень, які їм присвоюються у процесі виконання програми. Наприклад, коли текстовій величині `Edit1.Text` присвоюється числове значення (мал. 22.17).



Мал. 22.17

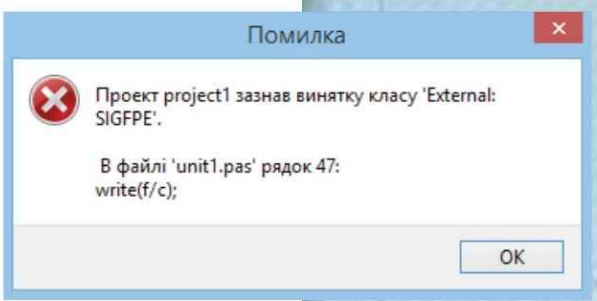
До помилок під час виконання належать помилки, пов'язані з неправильними числовими обчисленнями, помилки при знаходженні значень величин за формулами. Їх розпізнають тільки під час виконання програми. Наприклад, якщо у програмі мовою *Python* використати програмний код:

```
f=5
c=0
r=f/c
```

то після запуску програми на виконання у вікні виконання отримаємо повідомлення:

```
r=f/c
ZeroDivisionError^ division by zero
```

У середовищі *Lazarus* такі помилки називають винятками, вони супроводжуються виведенням на екран повідомлення про помилку (мал. 22.18).



Мал. 22.18

Логічні помилки — це помилки алгоритму, який лежить в основі програми. Результат, отриманий у ході виконання програми, не збігається з очікуваним результатом. Такі помилки не можна виявити засобами програмного середовища.

5. Як налагоджують програми в середовищах програмування?

Виправити синтаксичні помилки та помилки часу виконання програми можна після їх виявлення. А тоді знову запустити проект на виконання. Для того щоб перевірити наявність логічних помилок, потрібно запустити програму з тестовим (вхідним) набором даних і перевірити, чи збігається результат «ручного» підрахунку та програмного.

Процес пошуку логічних помилок у тексті програми з використанням тестових наборів вхідних даних називають **тестуванням програми**.

Допомагають у пошуку логічних помилок коментарі, які можна включати до тексту програми. **Коментарі** — це текст, який не виконується після запуску програми, а розміщується для пояснення структури програмного коду.

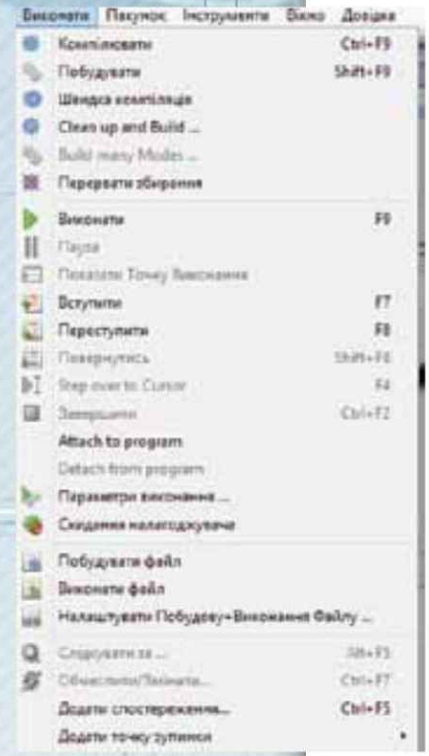
Мовою програмування *Python* коментарі розпочинаються із символу #, а мовою програмування *Free Pascal* — символами // — якщо коментар не займає більш ніж рядок, {} або (* *) — коментар, який може займати декілька рядків.

Якщо текст програми великий, то для пошуку місця логічної помилки інколи зручно тестувати програму не повністю, а частинами. Фрагменти програми, які тимчасово не потрібно використовувати у програмному коді, позначають як багаторядкові коментарі. Тоді після запуску проекту на виконання закоментований фрагмент програмного коду не буде виконуватись.

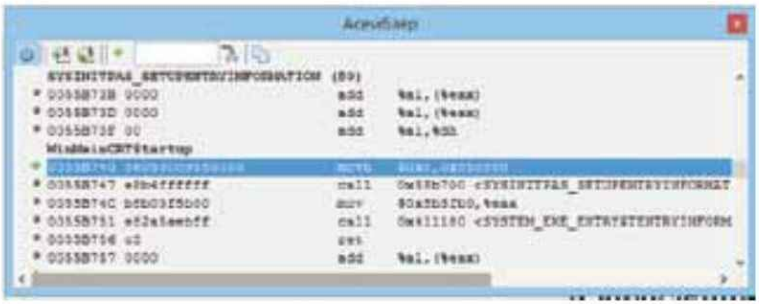
Ще одним способом виявлення у програмі логічних помилок є її покрокове виконання, або **трасування**. Крок виконання програми — це

Цікаво

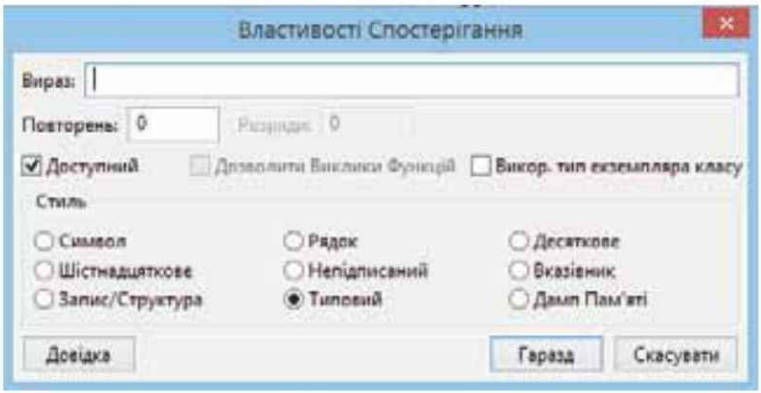
Тестовий набір даних — це значення змінних величин, що використовуються у програмному коді, для яких відомий результат виконання програми. Тестові набори даних у своїй роботі використовують тестувальники, які разом із програмістами беруть участь у розробці програмного забезпечення.



Мал. 22.19



Мал. 22.20



Мал. 22.21

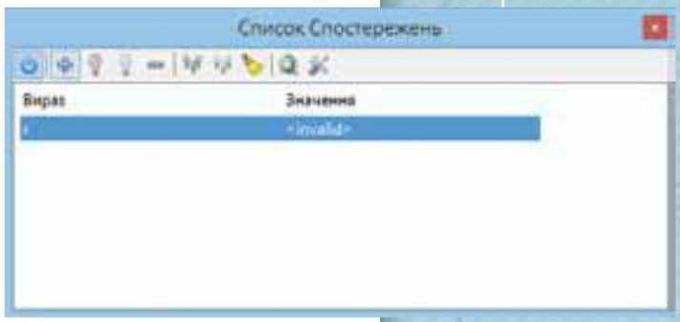
рядок програми. Для виконання одного кроку програми в середовищі *Lazarus* натискують клавішу *F7* або в меню *Виконати* обирають вказівку *Вступити* (мал. 22.19).

Після першого натиснення клавіші *F7* відбувається компіляція проекту, і якщо проект не містить синтаксичних помилок, то починається його виконання, при цьому з'являється вікно програми в машинних кодах (мал. 22.21).

З кожним наступним натисненням клавіші *F7* виконуватимуться команди наступного рядка: буде відкрито вікно програми, у яке можна ввести дані, змінним величинам буде присвоєно введені значення, виконані обчислення і т.д., насамкінець — виведений результат. Щоб під час виконання програми слідкувати за зміною значень змінних, налаштовують параметри вікна *Властивості Спостерігання*, виклик якого здійснюється вказівкою *Додати спостереження* меню *Виконати* або натискуванням клавіш *Ctrl+F5* (мал. 22. 21).

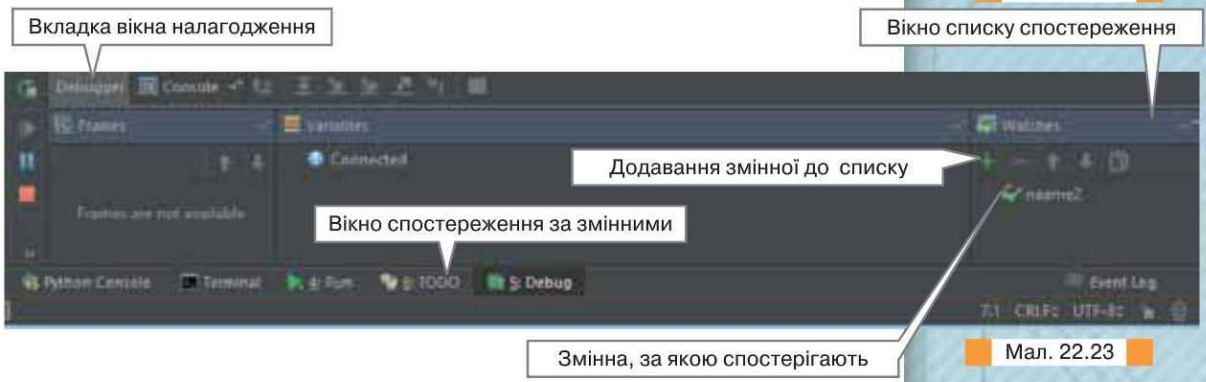
У полі *Вираз* записують ідентифікатори змінних, за якими будуть спостерігати в ході покрокового виконання програми. Значення введених змінних буде відображатись у вікні *Список спостережень*, яке з'являється після натиснення кнопки *Гаразд* (мал. 22.22).

Слідкуючи за значеннями змінної, за якою спостерігають, і порівнюючи їх з очікуваними, можна помітити, у який момент виконання програми це значення не відповідає очікуванням. Саме воно може бути причиною логічної помилки.



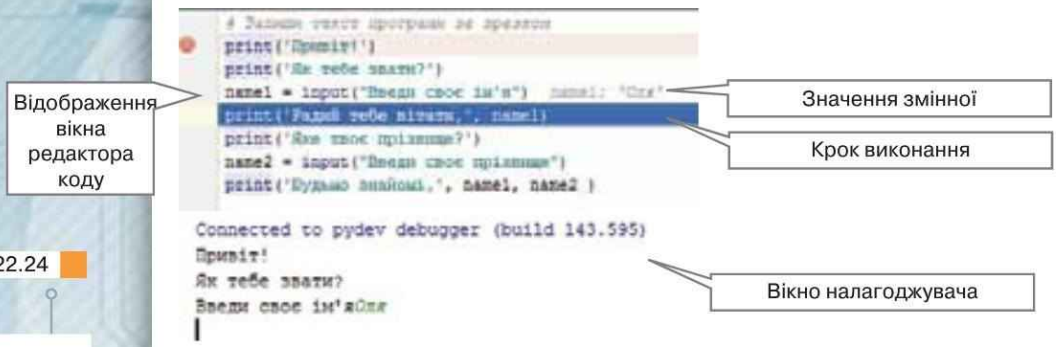
Мал. 22.22

У середовищі *PyCharm* налагодження програми відбувається за схожим сценарієм. У меню *Run* (*Виконати*) або на панелі інструментів обирають вказівку *Debug* (*Налагодити*) . У нижній частині вікна середовища відображається вікно налагодження (мал. 22.23).



Мал. 22.23

У редакторі коду відображається покроковий процес виконання програми та значення введених чи обчислених змінних, який змінюється з натисненням клавіші *F7* (мал. 22.24).



Мал. 22.24

Вказівки, з якими працюють під час налагодження програми, можна викликати в меню *Run* (*Запуск*) або натисненням відповідної комбінації клавіш.

ОБГОВОРЮЄМО

Обговоріть запитання, що містяться у файлі *Тема 21*, який зберігається у папці *Обговорюємо*. Також відкрити ці запитання можна за допомогою QR-коду.

ПРАЦЮЄМО В ПАРАХ

1. Обговоріть, як мовами програмування *Free Pascal* і *Python* записати введене слово у зворотному порядку. Складіть кожен відповідні програми в обраному середовищі програмування та порівняйте їх.
2. У програмі, написаній мовою програмування *Free Pascal* чи *Python,* використовуючи значення текстової змінної *s1,* отримали *s2* (табл. 22.9). Складіть відповідний програмний код. Перевірте, чи одинакові засоби отримання значення змінної *s2* із змінної *s1* у вибраній мові програмування використав кожний із вас. Зробіть висновки.

Таблиця 22.9

Значення змінної *s1*	**Значення змінної *s2***
Слово — не спіймаєш, вилетить —не горобець	Слово — не горобець, вилетить — не спіймаєш
Світнебездобрихлюдей	Світ не без добрих людей
Вік учись — вік живи	Вік живи — вік учись

3. Обговоріть, як у мовах програмування *Free Pascal* і *Python* можна вивести текст заяви про зарахування до гуртка технічної творчості у прийнятому для такого документа вигляді, у якій дані про заявника вносяться у процесі виконання програми. Визначте спільне та відмінності.
4. Обговоріть, у чому схожість процесу налагодження програми в середовищі програмування та редагування тексту в середовищі текстового процесора. Чи можна вважати редагування та налагодження синонімами?
5. Обговоріть і створіть засобами текстового процесора порівняльну таблицю процесу налагодження програми у проектах, створених у середовищах *Lazarus* і *PyCharm*.

ПРАЦЮЄМО САМОСТІЙНО

1. Складіть програмний код, який можна використати в середовищах *Lazasus* і *PyCharm*, щоб зі слова *s1* отримати слово *s2* (табл. 22.10).

Таблиця 22.10

№	Слово s1	Слово s2	№	Слово *s1*	Слово *s2*
1	рекомендація	оцінка	3	чиновник	новини
2	університет	турист	4	спортсмен	метро

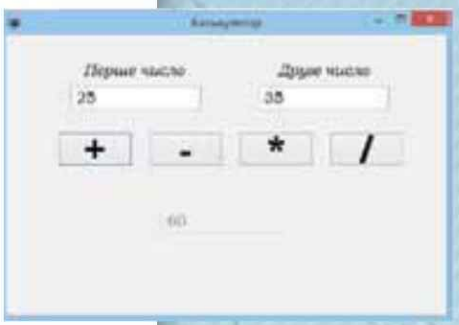
2. У самостійно вибраному середовищі програмування розробіть проект *Шифрувальник*, у якому в текстове поле вводять слово з п'яти літер, а після натиснення кнопки *Старт* у вікні повідомлення отримують результат за правилами, описаними в таблиці 22.11.

Таблиця 22.11

№	Результат	Введене значення	Отримане значення
1	Усі літери написані двічі	школа	шшккооллаа
2	Змінено порядок літер у парах	книга	нкгиа
3	Після кожної літери додано останню літеру слова	голуб	гбоблбубб
4	Кожна літера замінена на відповідний код у кодовій таблиці	зошит	10791086109610801090
5	Кожна літера замінена на наступну в кодовій таблиці	аркуш	бслфщ

3. Розробіть проект *Розрядні одиниці*, у якому в текстове поле вводять чотиризначне ціле число, натискають кнопку *Розкласти* й у текстовому полі, захищеному від змін, отримують запис числа у вигляді суми розрядних одиниць. Наприклад, для введеного числа *5843* отримують *5*1000+8*100+4*10+3*1.*

4. Розробіть проект *Калькулятор*, у якому в текстові поля вводять числа, після натиснення кнопки з позначкою математичної дії в текстовому полі, а яке захищене від змін, отримують результат (мал. 22.25).
Задайте тестові значення для перевірки роботи програми самостійно. Налагодьте програму засобами середовища *Lazarus*.



Мал. 22.25

5. Розробіть проект *Дата народження* в середовищі *PyCharm*, у якому в текстові поля користувач вводить день, місяць і рік свого народження, а після натиснення кнопки *Пуск* у відповідному текстовому написі формується дата народження записана у форматі дд/мм/рр. Чим буде відрізнятися цей проект від проекту, створеного в середовищі *Lazarus*?

23. ПРАКТИЧНА РОБОТА 11

НАЛАГОДЖЕННЯ ГОТОВОЇ ПРОГРАМИ

ПРИГАДАЙТЕ

- як співвідносяться типи даних та елементи для введення значень величин;
- які операції можна виконувати над числовими й текстовими величинами;
- як налагоджувати й виконувати програми в середовищі програмування;
- як аналізувати результати виконання програм.

СТВОРІТЬ

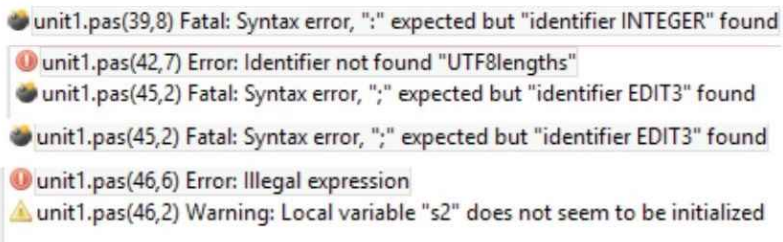
У власній структурі папок створіть папку *Практична робота 11.*

ПАМ'ЯТАЙТЕ

Під час виконання практичних завдань пам'ятайте про правила безпеки життєдіяльності при роботі з комп'ютером!

Завдання1. Перетворення тексту (*11 балів*)

Після запуску проекту *Перетворення тексту*, який реалізований у середовищі програмування *Lazarus*, у вікні повідомлення користувачі отримують відомості про синтаксичні помилки:



Відшукайте помилки у програмному коді та виправте їх. Запустіть проект на виконання та перевірте наявність логічних помилок у програмному коді.

Завдання 2. Вирази (*10 балів*)

Для обчислення значень виразів у середовищі програмування *PyCharm* розроблено проект *Вирази*. Реалізуйте проект для тестових значень, обраних самостійно, та порівняйте результат виконання програми з результатом, отриманим за допомогою інженерного режиму стандартної програми *Калькулятор*.

Завдання 3. Таблиця значень (*10 балів*)

У програмному коді проекту *Таблиця значень*, який реалізований у середовищі програмування *Lazarus* та збережений у папці *Програмування*, немає синтаксичних помилок. Доберіть тестові значення для перевірки на наявність помилок під час виконання. За потреби внесіть зміни у програмний код проекту.

24. ОПРАЦЮВАННЯ ВЕЛИЧИН ЛОГІЧНОГО ТИПУ. КОМАНДА РОЗГАЛУЖЕННЯ

ПРИГАДАЙТЕ:

- що таке висловлювання та як їх класифікують;
- правила використання алгоритмів із розгалуженням;
- опис алгоритмів із повним і неповним розгалуженням;
- опис умови в алгоритмах із розгалуженням у середовищі *Скретч*.

ВИ ДІЗНАЄТЕСЯ:

- як порівнюють значення величин у програмах;
- які операції можна виконувати над логічними величинами;
- як описати алгоритмічну структуру розгалуження мовами програмування;
- як за допомогою елементів управління задати логічне значення величини;
- для чого на формі використовують елемент управління *список, що розкривається*.

ВИВЧАЄМО

1. Як порівнюють значення величин у програмах?

Ви вже вмієте використовувати висловлювання для запису умов. Прості висловлювання мовами програмування можна записати у вигляді логічних виразів із використанням операцій порівняння (табл. 24.1).

Таблиця 24.1

Операція	Мова програмування	
	Free Pascal	*Python*
Більше	>	>
Менше	<	<
Не більше	<=	<=
Не менше	>=	>=
Дорівнює	=	==
Не дорівнює	<>	!=

Результатом виконання операції порівняння значень двох величин є величина **логічного типу**, яка може набувати одного із двох значень: `True` або `False`. Для опису логічних величин у мовах програмування *Free Pascal* та *Python* використовують службові слова (табл. 24.2).

Порівняння значень величин різних типів має певні особливості. Числові величини порівнюються за правилами математики. Порівняння текстових величин мовою *Free Pascal*, у результаті якого отримано логічне значення `True`, продемонстровано на малюнку 24.1.

```
'Алгоритм'<'алгоритм' – код великої літери А менший за код літери а;
'алгоритм' > 'аго' – довжина першої величини більша за довжину другої;
'алгоритм' <>'alhoritm'
'алг'='алг'.
```

Мал. 24.1

Цікаво

Логічний тип отримав свою назву на честь англійського математика та логіка середини XIX ст., одного із засновників математичної логіки Джорджа Буля.

Таблиця 24.2

Free Pascal	*Python*
boolean	*bool*

Для логічних величин a = `True`, b = `False` результатом виконання операції $a > b$ буде значення `True`, оскільки істинне значення `True` інтерпретується як 1, а хибне `False` — 0. Зрозуміло, що $1 > 0$.

2. Які операції виконують над логічними величинами?

Окрім операцій порівняння над логічними величинами в усіх мовах програмування для запису складених висловлювань використовують логічні операції:

- `not` (*не*) — заперечення;
- `and` (*і*) — кон'юнкція;
- `or` (*або*) — диз'юнкція (нестрога);
- `xor` (*виключне або*) — диз'юнкція (строга).

Результати виконання цих операцій над змінними A та B логічного типу, які набувають значень `True` (1) та `False` (0) наведені в **таблиці істинності** (табл. 24.3).

Таблиця 24.3

A	**B**	**not A**	**A and B**	**A or B**	**A xor B**
1	1	0	1	1	0
1	0	0	0	1	1
0	1	1	0	1	1
0	0	1	0	0	0

З таблиці істинності видно, що за допомогою логічної операції `not` змінюють значення логічної величини на протилежне. Результат операції `and` збігається з результатом множення значень 1 та 0 — є істинним лише за умови, що обидві величини A та B набувають значення `True` — істина. Результат операції *or* буде хибним лише за умови, що величини A і B набувають значення `False`, та істинним в усіх інших випадках. Наприклад, якщо $A = 5$, а $B = 7$, то значенням логічного виразу $(A < B)$ `and` $(B = 7)$ є `True`, оскільки логічні вирази $5 < 7$ та $B = 7$ набувають значення `True`, а тому за таблицею істинності результатом операції `and` є значення `True`.

Щоб визначити значення складеного логічного виразу, який містить кілька логічних операцій, використовують таблицю істинності та враховують пріоритет виконання операцій: у першу чергу виконується операція `not`, далі — `and`, в останню чергу — `or`. Як і для числових виразів, для зміни порядку виконання логічних операцій використовують дужки.

У мові програмування *Python* передбачено, що логічні значення *True* та *False* можна перетворювати на значення інших типів. Наприклад, при перетворенні їх у рядкові величини отримаємо слова `'True'` та `'False'` відповідно. Для перетворення логічних значень у значення числових типів використовують функції перетворення `int`, `float`. У цьому разі `True` відповідає одиниці, а `False` — нулю як для цілих, так і для дійсних чисел (мал. 24.2).

У зворотному порядку перетворення працює трохи по-іншому, але запам'ятати дуже просто: будь-які «непорожні» значення конвертуються в `True`, будь-які «нульові» — у `False`. Для примусового приведення значення до логічного типу використовується вбудована функція `bool` (мал. 24.3).

Цікаво

Операцію `or` ще називають логічним додаванням, а `and` — логічним множенням. Назви кон'юнкція та диз'юнкція походять відангл. *conjunction* — об'єднання та *disjunction* — роз'єднання. У математиці та логіці для запису логічних операцій прийняті спеціальні позначення. Операцію `and` позначають $\wedge$, операцію `or` — $\vee$.

```
int(True) == 1
int(False) == 0
float(True) == 1.0
float(False) == 0.0
str(True) == 'True'
str(False) == 'False'
```

Мал. 24.2

```
bool(None) == False
bool(1) == True
bool(-1.1) == True
```

Мал. 24.3

ДІЄМО

Вправа 1. Таблиця істинності для логічного виразу.
Завдання. Визначте, якого значення може набувати логічний вираз

(*not A*) *or* (*B and A*)

залежно від значень, яких набуватимуть логічні змінні *A* та *B*.

1. Обчислимо кількість можливих наборів значень логічних змінних за формулою $N = 2^n$, де N — кількість наборів значень, n — кількість змінних. У нашому випадку $N = 2^2 = 4$. Це число визначає кількість рядків у таблиці істинності.
2. Визначимо кількість і порядок логічних операцій у виразі.

```
      1     3      2
(not A) or (B and A)
```

Ця кількість і кількість логічних змінних визначає кількість стовпців у таблиці істинності. Для нашого випадку: 3 + 2 = 5.

3. Побудуємо таблицю істинності. Для зручності запису приймемо `True` = *1*, `False` = *0*. Для визначення значень кожної логічної операції використаємо таблицю істинності 24.3.

A	*B*	*not A*	*1*	*2*	*3*
			A and *B*	*A* or *B*	*A* xor *B*
1	1	0	1	1	0
1	0	0	0	1	1
0	1	1	0	1	1
0	0	1	0	0	0

4. Зробимо висновок: логічний вираз набуде значення `False` тільки тоді, коли змінна *A* набуде значення `True`, а змінна *B* — значення `False`. В усіх інших випадках логічний вираз матиме значення `True`.

3. Як описати алгоритмічну структуру розгалуження мовами програмування?

Прості та складені логічні вирази, значеннями яких є `True` або `False`, використовують в описі алгоритмічної структури розгалуження, за допомогою якої виконавець алгоритму може вибрати один зі сценаріїв подальших дій залежно від виконання певної умови.

Для опису алгоритмічної структури розгалуження мовою програмування, як і в середовищі *Скретч*, використовують **оператор неповного розгалуження** (табл. 24.4) та **оператор повного розгалуження** (табл. 24.5).

Таблиця 24.4

Мова програмування	Опис
Free Pascal	`if <логічний вираз>` `then <команда>;`
Python	`if <логічний вираз>:` `  блок команд`

```
begin
    <команда 1>;
   <команда 2>;
   …
   <команда n>;
end;
```

Мал. 24.4

Команда або блок команд в операторі неповного розгалуження буде виконуватися тільки у випадку, коли логічний вираз набуває значення `True`.

В операторі неповного розгалуження мовою програмування *Free Pascal* після службового слова `then` можна записати тільки одну команду або блок команд, які обмежені операторними дужками `begin, end` (мал. 24.4).

Мовою програмування *Python* в операторі неповного розгалуження після логічного виразу записують символ «:» і наступний рядок розпочинають із відступу в 4 символи. Усі команди, що мають такий відступ у програмному коді, входять до блоку команд, які будуть виконуватись у разі, якщо умова істинна.

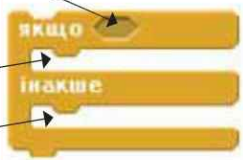
Для запису оператора повного розгалуження додатково використовують службове слово `else` (табл. 24.5).

Таблиця 24.5

Мова програмування	Опис

Free Pascal	`if <логічний вираз>`
	`then <команда 1>`
	`else <команда 2>;`

Python	`if <логічний вираз>:`
	`  блок команд 1`
	`else:`
	`  блок команд 2`



Якщо є потреба врахувати значення декількох логічних виразів, то використовують вкладені розгалуження (табл. 24.6).

Таблиця 24.6

Мова програмування	Опис

Free Pascal	`if <логічний вираз 1> then <команда 1>`
	`else if <логічний вираз 2> then <команда 2>`
	`…`
	`else if <логічний вираз n> then <команда n>`
	`else <команда інакше>;`

Python	`if <логічний вираз1>:`
	`  блок команд 1`
	`elif <логічний вираз2>:`
	`  блок команд 2`
	`...`
	`elif <логічний вираз n>:`
	`  блок команд n`
	`else:`
	`  <блок команд інакше>`

Мал. 24.5

В описаних вкладених розгалуженнях, якщо значення логічного виразу 1 — `True`, то виконується `команда 1` чи `блок команд 1`. Якщо зна-

чення логічного виразу 1 — `False` і значення логічного виразу 2 — `True`, то виконується `команда 2` чи `блок команд 2` і т.д. В іншому разі виконується команда `інакше` або `блок команд інакше` (мал. 24.5).

ДІЄМО

Вправа 2. Агрегатний стан води.

Завдання. Розробіть проект у середовищі *Lazarus*, у якому за введеним значенням температури води буде визначатись її агрегатний стан.

Мал. 24.6

1. Сплануйте проект. Передбачте, які об'єкти будуть використані на екранній формі та які події відбуватимуться з ними.
2. У папці *Навчальні проекти* власної структури папок створіть папку *Стан_води*.
3. Відкрийте середовище *Lazarus*, створіть новий проект і збережіть його складові в папку *Стан_води*. Змініть властивості об'єкта `Form1`, розмістіть на формі потрібні об'єкти та надайте значень їх властивостям, щоб отримати форму, як на малюнку 24.6.
   Візьміть до уваги, що для всіх об'єктів використовується властивість шрифта — `Arial Unicode MS`, розмір — 16. Стиль шрифта визначте за малюнком. Задайте значення `False` властивості `Enabled` текстового поля для виведення значення.
4. Створіть процедуру опрацювання події натиснення кнопки *Визначити*. У вікні редактора коду введіть програмний код (мал. 24.7). Зверніть увагу на структуру команди розгалуження.

```
procedure TForm1.Button1Click(Sender: TObject);
var t:integer;
begin
t:=StrToint(Edit1.Text);
if t<0 then Edit2.Text:='Вода перебуває у твердому стані'
    else if t>100 then Edit2.Text:='Вода перебуває у газоподібному стані'
            else Edit2.Text:='Вода перебуває у рідкому стані';
end;
```

Мал. 24.7

5. Запустіть проект на виконання. Введіть значення температури, наприклад, 25. Перевірте отриманий результат. Установіть, яка умова відповідає отриманому значенню.
6. Запустіть проект на виконання ще раз для значення –10,5. Поясніть, чому проект не виконується. Внесіть зміни до проекту так, щоб користувач міг задавати температуру води дійсним числом. Зауважте, що дійсне число, яке вводиться в текстове поле, записується із десятковою комою, а в програмному коді використовують число, записане з десятковою крапкою.
7. Збережіть зміни у проекті. Завершіть роботу із проектом і середовищем програмування.

Мал. 24.8

4. Як за допомогою елементів управління задати логічне значення величини?

Для реалізації розгалуження в проекті можна використати елементи управління: прапорець `CheckBox` або перемикач `RadioButton` (мал. 24.8).

Елементи управління CheckBox та RadioButton можуть набувати один із двох виглядів: увімкнений та не увімкнений. Тому за їх допомогою у програмний код можна передати логічне значення True — увімкнений, або False — не увімкнений.

На одній формі можна використати декілька увімкнених прапорців CheckBox, але тільки один увімкнений перемикач RadioButton.

Окрім стандартних властивостей, таких як Caption, Font та інших, які ви вже використовували для інших елементів управління, компонент CheckBox має особливі (табл. 24.7).

Таблиця 24.7

Властивість	Опис
Checked	Зміна стану прапорця: якщо значення цієї властивості *True,* то прапорець увімкнений; якщо значення властивості *False*, то прапорець не увімкнений (немає позначки ✓). Значення змінюється автоматично, хоча його також можна змінити у програмному коді
State	Стан прапорця. Дає змогу задавати стан увімкнення прапорця за одним із параметрів: увімкнений cbChecked, не увімкнений cbUnChecked, проміжний стан (сірий) cbGrayed

З елементом управління CheckBox пов'язані події OnChange та OnClick.

Подія OnClick виникає кожного разу, коли користувач вмикає чи вимикає прапорець на формі після запуску програми на виконання. Подія OnChange також виникає, коли користувач вмикає чи вимикає прапорець. На відміну від події OnClick, яка відбувається лише під час клацання на прапорці, подія OnChange відбувається в будь-якому разі, коли змінюється стан прапорця — якщо користувач мишею увімкнув-вимкнув прапорець, чи якщо така зміна передбачена у програмному коді, що виконується після натиснення деякої кнопки, за допомогою команди:

```
ChB1.Checked:= not ChB1.Checked;
```

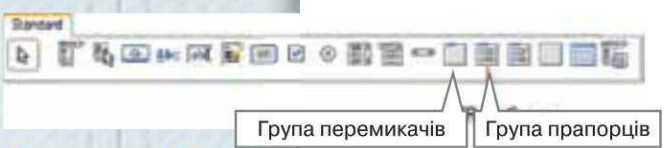
Тоді при натисненні на таку кнопку стан прапорця зміниться на протилежний — відбудеться OnChange , а подія OnClick — ні, адже мишею на прапорці не клацали.

Щоб додати до програмного коду процедуру опрацювання однієї з цих подій, можна двічі клацнути на ній у таблиці вікна *Інспектор об'єктів*.

Властивість Checked елемента управління RadioButton, яка визначає стан перемикача, не може бути змінена у програмному коді, на відміну від аналогічної в елемента управління CheckBox.

Крім компонентів CheckBox та RadioButton, на форму можна додати компоненти CheckGroup або RadioGroup (мал. 24.9). Їх використовують у разі, якщо необхідно розмістити декілька груп прапорців чи перемикачів. Якщо на формі використано декілька груп перемикачів, у межах кожної з груп можна увімкнути по одному.

Із перемикачами та прапорцями в межах групи виконують ті самі дії, як і з окремими елементами управління на формі.



Мал. 24.9

Елементи управління `RadioGroup` і `CheckGroup` мають свої особливі властивості (табл. 24.8).

Таблиця 24.8

Властивість	Опис
`Caption`	Заголовок групи
`Columns`	Кількість стовпців елементів у групі. За замовчуванням — 1
`ItemIndex`	Визначає номер (починаючи з 0) елемента управління, який виділений у групі. Якщо не виділений жоден, то значення властивості дорівнює –1
`Items`	Містить список заголовків елементів групи. Для введення заголовків відкривається редактор, який викликають за допомогою кнопки [...], розташованої праворуч у рядку властивості *Items*

Щоб додати елемент управління *прапорець* у програмі, написаній мовою *Python* у середовищі *PyCharm*, після підключення модуля графічного інтерфейсу `tkinter` та елемента управління *прапорець* `from tkinter import ttk`, слід описати об'єкт *прапорець*:

```
bool_var = tkinter.BooleanVar()
```

Потім слід розмістити прапорець на головній формі `main`:

```
check = ttk.Checkbutton(main, text="Напис", variable=bool_var)
check.pack()
```

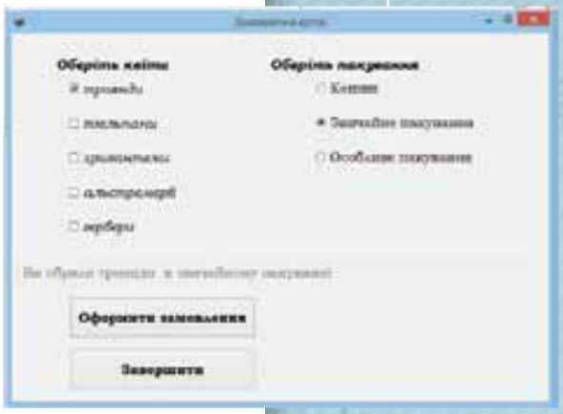
Для перевірки, чи прапорець увімкнено, використовують команду

```
bool_var.get()
```

ДІЄМО

Вправа 3. Замовлення квітів.

Завдання. Складіть проект *Замовлення квітів* у середовищі *Lazarus* для оформлення електронного замовлення квітів за зразком (мал. 24.10). У проекті натиснення кнопки *Оформити замовлення* викликає виведення відповідного тексту в текстовому полі, яке захищене від змін, а натиснення кнопки *Завершити* — закриває вікно форми.



Мал. 24.10

1. Сплануйте проект. Передбачте, які об'єкти будуть використані на екранній формі та які події будуть відбуватись із ними.
2. У папці *Навчальні проекти* власної структури папок створіть папку *Замовлення_квітів*.
3. Відкрийте середовище *Lazarus*, створіть новий проект і збережіть його складові в папку *Замовлення_квітів*. Змініть властивості об'єкта `Form1`, розмістіть на формі потрібні об'єкти та надайте значень їх властивостям, щоб отримати вигляд, як на малюнку 24.7. Візьміть до уваги, що для всіх об'єктів використовується властивість шрифта — *Bookman Old Style*, розмір — 14. Стиль шрифта визначте за малюнком.
4. Створіть процедуру опрацювання події: натиснуто кнопку *Оформити замовлення*. У вікні редактора коду введіть код (мал. 24.11).
5. Створіть процедуру опрацювання події натиснення кнопки *Завершити*. Для цього використайте метод `Close`.

```
procedure TForm.Button1Click(Sender: TObject);
var k1,k2,k3,k4,k5,p1,p2,p3: boolean;
      s, t: string;
begin
      k1:= CheckBox1.checked;
      k2:= CheckBox2.checked;
      k3:= CheckBox3.checked;
      k4:= CheckBox4.checked;
      k5:= CheckBox5.checked;
      p1:= RarioButton1.checked;
      p2:= RarioButton2.checked;
      p3:= RarioButton3.checked;
      s:='';
      if k1 then s:=s+CheckBox1.Caption+' ';
      if k2 then s:=s+CheckBox2.Caption+' ';
      if k3 then s:=s+CheckBox3.Caption+' ';
      if k4 then s:=s+CheckBox4.Caption+' ';
      if k5 then s:=s+CheckBox5.Caption+' ';
      if p1 then t:=' в кошику';
      if p2 then t:=' в звичайному пакуванні;
      if p3 then t:=' в особливому пакуванні;
      Edit1.Text:='Ви обрали '+s+t;

end;
```

Змінні *k1, k2, k3, k4, k5* визначають стан елементів управління `CheckBox`

Змінні *p1, p2, p3* визначають стан елементів управління `RadioButton`

У текстовій змінній *s* формується повідомлення про вибір квітів

У текстовій змінній *t* формується повідомлення про пакування

Виведення результату в текстове поле

Мал. 24.11

6. Запустіть проект на виконання. Перевірте, чи відповідають умові завдання дії, пов'язані з об'єктами управління екранної форми. Якщо є помилки, — виправте їх.
7. Завершіть роботу із проектом і середовищем програмування.

Вправа 4. Кольорова форма.

Завдання. Розробіть проект *Кольорова форма* в середовищі *Lazarus*, у якому форма змінюватиме свої властивості — колір і розмір залежно від обраних перемикачів у групах *Колір* та *Розмір*.

1. У папці *Навчальні проекти* власної структури папок створіть папку *Кольорова_форма*.
2. Відкрийте середовище *Lazarus*, створіть новий проект і збережіть його складові в папку *Кольорова_форма*. Змініть значення властивості `Caption` об'єкта `Form1` на *Кольори.*

Мал. 24.12

3. Розмістіть на екранній формі потрібні об'єкти — групи перемикачів із заголовками *Колір* та *Розмір* (мал. 24.12).
4. Змініть властивість `Items` кожної групи, увівши відповідні пояснення до перемикачів (мал. 24.13).
5. Створіть процедуру опрацювання події: *натиснуто кнопку Пуск.*

У редакторі коду введіть команди для зміни значення властивості `color` екранної форми залежно від увімкненого перемикача. Значення цієї властивості розпочинається із літер `cl`, після яких записують англійською мовою назву кольору. Наприклад:

```
if radioGroup1.ItemIndex=0 then form1.color:=clred;
```

6. Введіть команди зміни розміру форми за зразком:

```
if radioGroup2.ItemIndex=0 then
   begin
     form1.height:=100;
     form1.width:=400;
   end;
```

7. Для середнього розміру оберіть висоту 200, ширину 600, а для великого — 300, 800 відповідно.
8. Запустіть проект на виконання. Перевірте, чи відповідають умові завдання дії, пов'язані з об'єктами управління екранної форми. Якщо є помилки — виправте їх.
9. Завершіть роботу з проектом і середовищем програмування.

Вправа 5. Слово чи декілька.

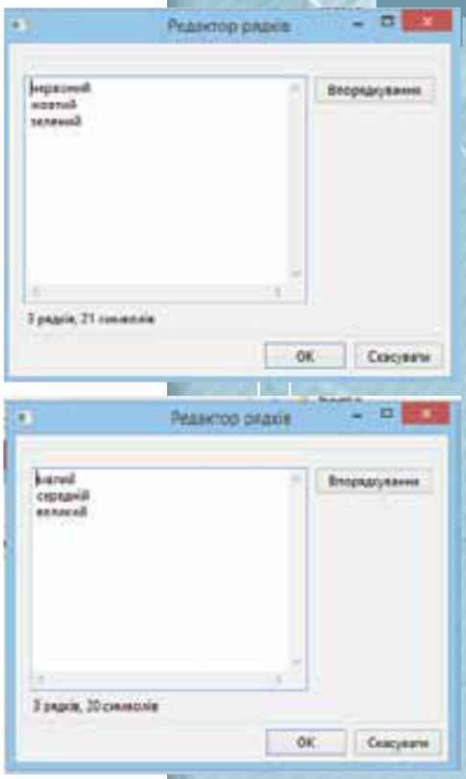
Завдання. Складіть у середовищі *PyCharm* проект, за допомогою якого можна визначити, чи введений текст складається з одного слова, чи з декількох.

1. Відкрийте середовище програмування *PyCharm*.
2. Створіть новий файл програми мовою *Python* з іменем *Слова* в папці *Навчальні проекти* власної структури папок.
3. У програмному коді запишіть команди за зразком (мал. 24.14).
4. Запустіть проект на виконання. Доберіть тестові дані та перевірте правильність складеного програмного коду.
5. Змініть програмний код так, щоб текст можна було вводити в текстове поле, а у вікні повідомлення — отримувати відповідь.

   Поясніть, які команди із запропонованих ви будете використовувати у програмному коді:

```
import tkinter
import tkinter.messagebox
main = tkinter.Tk()
str_var = tkinter.StringVar()
def button_click():
   s = str_var.get()
   if ' ' in s:
         tkinter.messagebox.showinfo("Результат",
         "введено декілька слів")
   else:
         tkinter.messagebox.showinfo("Результат",
         "введено одне слово")
label = tkinter.Label(text="Введіть текст")
label.pack()
edit = tkinter.Entry(main, textvariable = str_var)
edit.pack()
button = tkinter.Button(main, text="Перевірити",
command=button_click)
button.pack()
main.mainloop()
```

6. Запустіть проект на виконання. Отримайте результат для введеного одного слова та декількох. Завершіть роботу із середовищем.

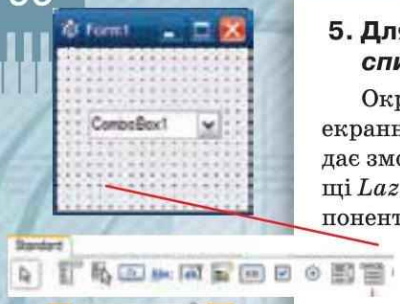


Мал. 24.13

Перевірка умови: пропуск входить у рядок *s*

```
s=input('введіть текст')
if ' ' in s:
    print('введено декілька слів')
else:
    print('введено одне слово')
```

Мал. 24.14



Мал. 24.15

5. Для чого на формі використовують елемент управління *список, що розкривається*?

Окрім текстового поля, яке може містити тільки одне значення, на екранних формах розміщують також елемент управління *список*, який дає змогу обрати одне значення із запропонованого переліку. У середовищі *Lazarus* створити список, що розкривається, можна за допомогою компонента `ComboBox` (мал. 24.15).

На відміну від інших елементів управління, які ви уже вивчали, список `ComboBox` має особливі властивості (табл. 24.9).

Таблиця 24.9

Властивість	Опис
`Items`	Елементи списку
`Count`	Кількість елементів списку
`Sorted`	Ознака необхідного сортування (якщо ця властивість має значення `True`) після додавання чергового елемента списку
`ItemIndex`	Номер вибраного елемента (нумерація розпочинається з нуля. Якщо жоден із елементів не вибраний — то значення дорівнює –1)
`DropDownCount`	Кількість елементів, які відображаються у списку. Щоб відобразити решту — використовують смугу прокручування

У програмах мовою *Python* у середовищі *PyCharm* для того, щоб використати список, що розкривається, після підключення модуля графічного інтерфейсу `tkinter` та елемента управління `from tkinter import ttk`, спочатку описують об'єкт *список*:

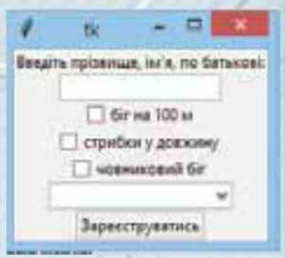
```
choice_var = tkinter.StringVar()
```

Потім розміщують список на головній формі `main`, надаючи значення елементам списку:

```
choice = ttk.Combobox(main, textvariable=choice_var)
choice['values'] = ('Значення1', 'Значення2', 'Значення3')
choice.pack()
```

Для визначення, який елемент списку обрано, використовують команду

```
choice _var.get()
```



Мал. 24.16

Вправа 6. Картка учасника змагань.

Завдання. Розробіть проект *Картка учасника змагань* у середовищі *PyCharm*, у якому користувач вводить своє прізвище, ім'я та по батькові в текстове поле, позначає прапорцями вид змагання, зі списку, що розкривається, обирає вікову групу. Після натиснення кнопки *Зареєструватись* (мал. 24.16) отримує у вікні повідомлення про підтвердження реєстрації.

1. Відкрийте середовище програмування *PyCharm*.
2. Створіть новий файл програми мовою *Python* з іменем *Картка* в папці *Навчальні проекти* власної структури папок.
3. Сплануйте, які об'єкти потрібно імпортувати до програми. Оберіть їх із запропонованого списку та додайте до проекту у вікні редактора коду:

```
import tkinter
```

```
from tkinter import ttk
from tkinter import messagebox
```

4. Визначте, які змінні із запропонованого списку можуть набувати значень *прізвище, ім'я та по батькові, вид змагання, вікова група*:

```
name_var = tkinter.StringVar()
bool_var1 = tkinter.BooleanVar()
bool_var2 = tkinter.BooleanVar()
bool_var3 = tkinter.BooleanVar()
choice_var = tkinter.StringVar()
```

5. Створіть об'єкти та розмістіть їх на головній формі за зразком (мал. 24.12). Для створення об'єкта *список, що розкривається*, запишіть команду:

```
choice['values'] = ('молодша', 'середня', 'старша')
```

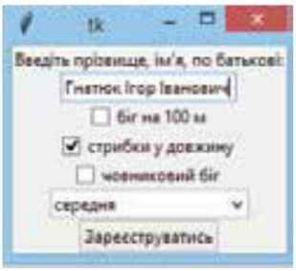
6. Запишіть код опрацювання події натиснення кнопки:

```
def button_click():
    s = ""
    s += name_var.get()+", ви обрали: "
    if bool_var1.get():
        s += "біг на 100 м "
    if bool_var1.get():
        s += "стрибки у довжину "
    if bool_var1.get():
        s += "човниковий біг "
        s += " у віковій категорії " + choice_var.get()
    messagebox.showwarning("Реєстрація", s)
```

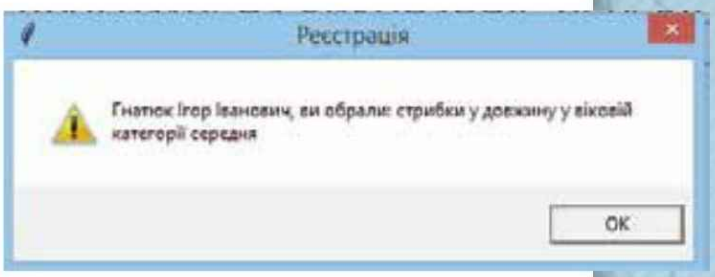
7. Додайте до програмного коду команду запуску опрацювання подій програми:

```
main.mainloop()
```

8. Запустіть програму на виконання, перевірте, чи для запропонованих значень (мал. 24.17, *а*) ви отримали очікуваний результат (мал. 24.17, *б*).



Мал. 24.17, *а*



Мал. 24.17, *б*

9. Збережіть зміни у проекті. Закрийте вікно середовища програмування.

ОБГОВОРЮЄМО

Обговоріть запитання, що містяться у файлі *Тема 24*, який зберігається у папці *Обговорюємо*. Також відкрити ці запитання можна за допомогою QR-коду.

ПРАЦЮЄМО В ПАРАХ

1. Складіть у парі п'ять виразів на порівняння значень величин різних типів вибраною мовою програмування. Запропонуйте їх іншій парі. Перевірте правильність отриманих відповідей.
2. Обговоріть, чим буде відрізнятись порядок розробки проекту про агрегатний стан води в середовищі *PyCharm* від його створення в середовищі *Lazarus*. Реалізуйте цей проект.
3. Обговоріть, чи достатньо відомих вам засобів мови програмування *Free Pascal* для того, щоб визначити, із скількох слів складається деяке речення, яке вводять у текстове поле після запуску проекту на виконання. Спробуйте реалізувати цей проект у середовищі *Lazarus*. Які переваги мови програмування *Python* над мовою програмування *Free Pascal* ви побачили при цьому?
4. Обговоріть, чи можна реалізувати проект *Картка учасника змагань* у середовищі *Lazarus*. Визначте, що доцільніше використовувати при розробці екранної форми — відповідні елементи управління чи їх групи? Розробіть проект, розподіливши ролі: один створює об'єкти екранної форми, а інший — пише програмний код опрацювання подій.

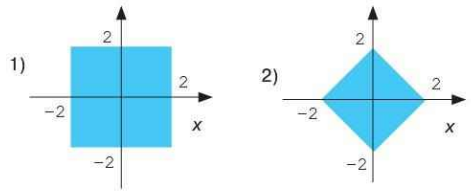
ПРАЦЮЄМО САМОСТІЙНО

1. Визначте, якого логічного значення набуває логічний вираз, записаний мовами програмування *Free Pascal* та *Python*:

Free Pascal	***Python***
1) `True and True`	1) `False and False`
2) `1 = 1 or 2 <> 1`	2) `1 == 1 and 2 == 1`
3) `'test' = 'test'`	3) `«test» == «testing»`
4) `'testing' <> 'test'`	4) `«test» != «testing»`
5) `False and 1 = 1`	5) `True and 0 != 0`
6) `not (10 = 1 or 1000 = 1000)`	6) `not (1 == 1 and 0 != 1)`

2. За допомогою умовного оператора мовами програмування *Python* та *Free Pascal* запишіть фрагмент програмного коду для:
   1) заміни значення змінної *s* дійсного типу його абсолютною величиною;
   2) присвоєння змінній *k* значення 0, якщо її початкове значення належало інтервалу (0;5);
   3) надання змінній *t* значення `True`, якщо введені координати x та y належать першій або третій координатній чверті;
   4) знаходження значення виразів $max^2(x,y) - min^2(x,y)$, де max — більше та min — менше з двох чисел x та y.
3. У вибраному самостійно середовищі програмування розробіть проект *Мішень*, у якому за введеними в текстові поля дійсними числами x та y (мал. 24.18) у вікні повідомлення буде виведено повідомлення «Вітаємо! Влучний постріл!», якщо точка з координатами: $(x; y)$ належить заштрихо-

ваній області, та повідомлення «Шкода, не влучив!» — в іншому разі.



Мал. 24.18

4. У вибраному самостійно середовищі програмування розробіть проект *Оцінки*, за яким у текстове поле виводиться опис досягнутого рівня на основі шкільної оцінки, яка обирається з елемента управління (оберіть самостійно — прапорці, перемикачі, список, що розкривається):

1) 1, 2, 3 — початковий рівень;
2) 4, 5, 6 — середній рівень;
3) 7, 8, 9 — достатній рівень;
4) 10, 11, 12 — високий рівень.

5. У вибраному самостійно середовищі програмування розробіть проект *Правила дорожнього руху*, за яким можна визначити, чи не порушив правила дорожнього руху водій, який проїхав відстань s за час t, якщо на шляху встановлено одне із запропонованих обмежень на швидкість: 40 км/год, 60 км/год, 90 км/год. Передбачте, що дані вводяться в текстові поля, обмеження обираються за допомогою елементів управління (оберіть самостійно), а результат — виводиться у вікні повідомлення.

6. У вибраному самостійно середовищі програмування розробіть проект *Килимки*, за яким можна визначити, яку максимальну кількість квадратних килимків зі стороною c, де c — ціле число, потрібно використати, щоб застелити кімнату з підлогою розміром $a \times b$, де a і b — цілі числа. У проекті також слід визначити, яка площа не буде накрита килимками, якщо їх не можна накладати чи підгинати. Передбачте також ситуацію, коли розмір килимка перевищує розмір підлоги кімнати.

7. Відомо таке співвідношення ідеальної ваги та зросту людини за віком: від значення зросту людини в сантиметрах беруться останні дві цифри; якщо вік людини до 25 років, то її вага має становити на 5 кг менше від одержаного двоцифрового числа, для людей від 25 до 45 років — дорівнювати цьому двоцифровому числу, а для старших за 45 років — двоцифрове число потрібно збільшити на 5. У вибраному самостійно середовищі програмування розробіть проект *Ідеальна вага*, у якому користувач вводить у текстове поле свій зріст у сантиметрах та вагу, обирає діапазон віку, до якого він належить. Після натиснення кнопки запуску проекту в текстовому полі чи вікні повідомлення отримують висновок: вага ідеальна, надлишкова вага, недостатня вага.

8. У середовищі програмування *Lazarus* розробіть проект *Веселка*, у якому після вибору кольору веселки та натиснення кнопки *Показати*, відкривається допоміжна екранна форма, колір якої відповідає обраному на головній формі.

9. У середовищі програмування *PyCharm* розробіть проект, у якому виконується перевірка, чи дане слово є паліндромом (можна прочитати зліва направо так само, як справа наліво). Наприклад, слово «око» — це паліндром. Передбачте засоби, які слід використати для того, щоб слово «Пилип» теж було визначене як паліндром, ураховуючи різницю між великими та малими літерами.

25. РЕАЛІЗАЦІЯ АЛГОРИТМІВ ПОВТОРЕННЯ МОВАМИ ПРОГРАМУВАННЯ

ПРИГАДАЙТЕ:

- з якою метою в алгоритмах використовують структуру повторення;
- подання алгоритмічної структури повторення графічним способом;
- опис алгоритмів із визначеною кількістю повторень у середовищі *Скретч*;
- опис алгоритмів повторення з умовою в середовищі *Скретч*;
- що таке вкладені повторення.

ВИ ДІЗНАЄТЕСЯ:

- як мовами програмування описують цикл із передумовою;
- як мовами програмування описують цикл із лічильником;
- які особливості використання циклів у програмах, описаних мовами програмування *Free Pascal* та *Python*;
- як задати випадкове число.

ВИВЧАЄМО

1. Як мовами програмування описують цикл із передумовою?

Ви знаєте, що алгоритм, у якому передбачається багаторазове виконання одного й того самого набору команд, називають **циклічним**. У циклічних алгоритмах використовують алгоритмічну структуру **повторення**. У мовах програмування для опису структури повторення використовують **оператори циклу.**

Будь-який оператор циклу складається з двох частин: **заголовка** та **тіла**. У заголовку циклу записуються умови, за яких виконання циклу триватиме або завершиться, а в тілі циклу містяться команди, виконання яких потрібно повторювати (мал. 25.1).

Мал. 25.1

У мовах програмування *Free Pascal* та *Python* цикл із передумовою описується оператором `while...` (табл. 25.1).

Таблиця 25.1

Мова програмування	Опис	
Free Pascal	**while** <логічний вираз> **do** <команда>;	Скорочена форма

	while <логічний вираз> **do**
	begin
	<команда1>;
	<команда2>;
	…
	<команда n>;
	end; (тіло циклу)
		Повна форма

Продовження таблиці 25.1

Мова програмування	Опис	
Python	`while <логічний вираз>:` `  <команда>`	Скорочена форма

	`while <логічний вираз>:`
	`  <команда1>`
	`  <команда2>`
	`  …`
	`  <команда n>`
	`else:`
	`  <команда інакше>`
	(тіло циклу)
		Повна форма

Скорочена форма оператора `while` працює таким чином. Визначається значення логічного виразу. Якщо це значення `True`, то виконується команда, записана після службового слова `do` (*Free Pascal*), або команда, виділена відступами, що входить до блоку `while` (*Python*). Якщо значення логічного виразу `False`, то цикл завершується й управління передається команді, записаній у програмному коді після тіла циклу. У тілі циклу обов'язково використовують команду, яка змінюватиме значення величини, що використовується в логічному виразі.

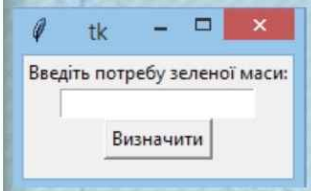
Повна форма оператора `while` мовою програмування *Free Pascal* передбачає, що після службового слова `do` будуть виконуватись команди, які містяться між операторними дужками `begin`... `end`. А мовою програмування *Python*, окрім команд, що входять до блоку `while`, використовують конструкцію `else`:, після якої в новому рядку роблять відступ 4 символи та записують команди, які мають виконуватись, якщо значення логічного виразу `False`.

ДІЄМО

Вправа 1. Урожай.

Завдання. Фермер вирощує новий сорт рослин для годівлі тварин. Він дослідив закономірність, що після кожного зрізу, який здійснюється через тиждень не більше 5 разів, кількість зеленої маси збільшується за формулою: $s = s + (i - 1) \cdot i$, де i — номер тижня, s — кількість зеленої маси. Розробіть проект у середовищі *PyCharm*, за яким буде визначено, чи встигне фермер зібрати n одиниць зеленої маси.

1. Відкрийте середовище програмування *PyCharm*.
2. Створіть новий файл програми мовою *Python* з іменем *Урожай* у папці *Навчальні проекти* власної структури папок.
3. Імпортуйте об'єкти, які потрібні для опису графічного інтерфейсу програми (мал. 25.2).
4. Опишіть об'єкти, які ви будете використовувати у проекті.
5. Запишіть програмний код опрацювання події натиснення кнопки *Визначити*.



Мал. 25.2

```
def button_click():
        s = 1
        i=1

        st = str_var.get()
        k = int(st)
        while i <= 10:
            s += (i-1) * i
            i+=2

        if k <= s:
            tkinter.messagebox.showinfo("Результат", 'Встигне
            отримати '+ str(s))
        else:
            tkinter.messagebox.showinfo("Результат", 'Не встигне
            отримати ')
```

6. Запишіть програмний код розміщення об'єктів на головній формі:

```
label = tkinter.Label(text='Введіть потребу зеленої
маси:')
label.pack()
edit = tkinter.Entry(main, textvariable=str_var)
edit.pack()

button = tkinter.Button(main, text="Визначити",
command=button_click)
button.pack()
```

7. Запишіть програмний код запуску подій головної форми.
8. Запустіть проект на виконання. Перевірте, чи отримаєте ви для значення 50 повідомлення «Встигне отримати 121».
9. Завершіть роботу з проектом і середовищем.

Мал. 25.3

2. Як мовами програмування описують цикл із лічильником?

Для того щоб описати повторення визначену кількість разів, використовують оператор **циклу з лічильником** (мал. 25.3).

У мовах програмування *Free Pascal* та *Python* цикл із лічильником описується оператором `for`... (табл. 25.2).

Таблиця 25.2

Мова програмування	Опис

Free Pascal	параметр циклу; початкове значення параметра циклу; кінцеве значення параметра
	`for i:= <in> to <ik> do <команда>;`
	тіло циклу; крок зміни параметра циклу = 1
	`for i:= <in> downto <ik> do <команда>;`
	крок зміни параметра циклу = –1

Продовження таблиці 25.2

Мова програмування	Опис

Python	параметр циклу
	`for i in <діапазон>:`
	`<команда>`
	тіло циклу

Величина параметра циклу, його початкового та кінцевого значень можуть бути цілими числами або належати до деякого списку.

Крок зміни циклу завжди однаковий і дорівнює інтервалу між двома найближчими значеннями типу параметра (при цілочисельному значенні параметра крок дорівнює 1).

Для визначення діапазону значень параметра у програмах мовою програмування *Python* можна скористатись функцією `range` (табл. 25.3).

Таблиця 25.3

Опис	Значення
`range(6)`	0, 1, 2, 3, 4, 5
`range(3,8)`	3, 4, 5, 6, 7
`range(7,16,2)`	7, 9, 11, 13, 15

Можна скористатись величиною перелічуваного типу. Наприклад, списком непарних чисел першого десятка: 1, 3, 5, 7, 9, чи списком голосних літер українського алфавіту ‘а’,’о’,’у’, ‘е’,’и’,’і’, ’ї’, ’ю’. Список значень у мові програмування *Free Pascal* обмежують круглими дужками (), а в *Python* — квадратними [].

Якщо тіло циклу складається більш ніж з однієї команди, як і в циклі з передумовою, у мові програмування *Free Pascal* використовують операторні дужки `begin… end`.

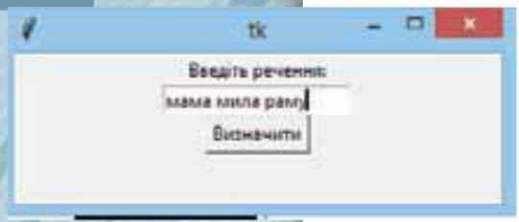
Цикл `for…` виконується за таким алгоритмом:

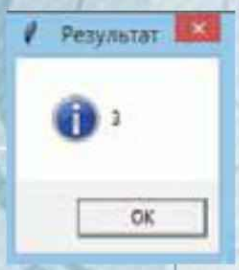
1. Параметру циклу `i` присвоюється початкове значення.
2. Якщо значення параметра циклу більше, ніж його кінцеве значення, то цикл завершується (у випадку циклу зі службовим словом `downto` мовою програмування *Free Pascal* цикл завершується, коли значення параметра цикла менше, ніж його кінцеве значення). В іншому разі виконується п. 3.
3. Виконується команда.
4. Значення параметра циклу `i` змінюється на відповідний крок і здійснюється перехід до п. 2. і т.д.

Таким чином, на відміну від оператора циклу `while`, в операторі циклу з лічильником зміна значення лічильника здійснюється автоматично.

ДІЄМО

Вправа 2. Кількість слів у реченні.
Завдання. Розробіть у середовищі *PyCharm* проект, за допомогою якого можна визначити, скільки слів введено в текстове поле, якщо відомо, що між словами міститься лише один пропуск.





Мал. 25.4

1. Відкрийте середовище програмування *PyCharm*.
2. Створіть новий файл програми мовою *Python* з іменем *Кількість_слів* у папці *Навчальні проекти* власної структури папок.
3. Імпортуйте об'єкти, які потрібні для опису графічного інтерфейсу програми (мал. 25.4).
4. Оголосіть та опишіть величини `main` та `str_var`.
5. Запишіть програмний код опрацювання події натиснення на кнопку *Визначити:*

```
def button_click():
    s = 0
    z = ' '
    st = str_var.get()
    k = len(st)
    for i in range(0, k):
      if z == st[i]:
        s = s + 1
    tkinter.messagebox.showinfo("Результат", str(s+1))
```

6. Запишіть у програмному коді команди для створення об'єктів на формі `main` та отримання значень змінних.
7. Запишіть команду запуску подій на формі.
8. Запустіть проект на виконання. Перевірте, чи результат для введеного речення відповідає малюнку 25.4.
9. Завершіть роботу з проектом і середовищем.

3. Які особливості використання циклів у програмах, описаних мовами програмування *Free Pascal* та *Python*?

У мові програмування *Free Pascal* реалізовано ще один цикл — **цикл із післяумовою:**

```
repeat
   <команди>;
until <логічний вираз припинення циклу>;
```

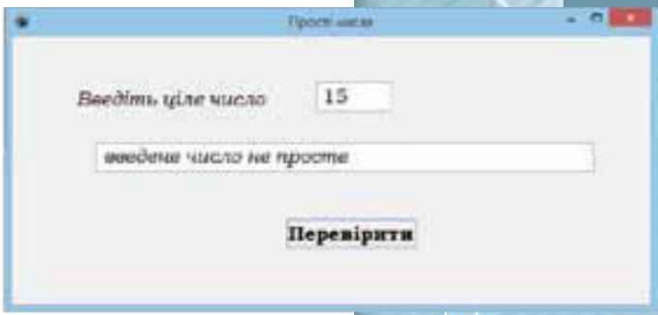
Якщо в циклі з передумовою перевірка істинності умови здійснюється до виконання команд у тілі циклу, то в циклі з післяумовою істинність умови перевіряється після виконання команд тіла циклу. Якщо значення логічного виразу `False`, то команди циклу повторюються, якщо `True`, то виконання циклу припиняється. Таким чином, цикл із післяумовою буде виконаний хоча б один раз, на відміну від циклу з передумовою, який може бути не виконаний жодного разу, якщо логічний вираз одразу має значення `False`. Якщо в тілі циклу потрібно використати більш ніж одну команду, то операторних дужок використовувати не потрібно — їх роль виконують службові слова `repeat` та `until`, які входять до конструкції оператора циклу.

У мові програмування *Python* можна використовувати достроковий вихід із циклу. За допомогою функції `break` можна перервати виконання команд тіла циклу та перевести управління до команди, яка слідує після блоку команд, що належать до `while` або `for`.

ДІЄМО

Вправа 3. Просте число.

Завдання. Число називають простим, якщо воно ділиться без остачі тільки на одиницю й на себе. Розробіть програмний проект у середовищі *Lazarus*, за допомогою якого буде перевірятись, чи є простим введене в текстове поле число (мал. 25.5).



Мал. 25.5

1. Сплануйте проект. Передбачте, які об'єкти будуть використані на екранній формі та які події відбуватимуться з ними.
2. У папці *Навчальні проекти* власної структури папок створіть папку *Просте_число*.
3. Відкрийте середовище *Lazarus*, створіть новий проект і збережіть його складові в папку *Просте_число*. Змініть значення властивостей об'єкта `Form1`, розмістіть на формі потрібні об'єкти та надайте значень їх властивостям, щоб отримати форму, як на малюнку 25.5.
4. У програмному коді опрацювання події натиснення кнопки *Перевірити* введіть команди (мал. 25.6).

```
var n, i: integer;
begin
n:=strToint(edit1.Text);
i:=1;
repeat
     i:= i+1
until n mod i=0;
if n=i then edit2.text:= 'введене число просте'
       else edit2.text:= 'введене число не просте';
end;
```

Мал. 25.6

5. Запустіть проект на виконання. Введіть число 15. Перевірте отриманий результат.
6. Запустіть проект на виконання ще раз для значення 19. Перевірте отриманий результат.
7. Збережіть зміни у проекті. Завершіть роботу з проектом і середовищем програмування.

4. Як задати випадкове число?

Складаючи алгоритми в навчальному середовищі *Скретч*, ви використовували команду, за допомогою якої отримували для значень параметрів деяких команд випадкові числа: координати розміщення об'єкта на сцені, кольору малюнка тощо (мал. 25.7).

Мал. 25.7

Випадкові значення величин часто використовують при складанні програм, зокрема ігрових.

Для того щоб у програмі мовою програмування *Python* використати значення, яке передається у програмний код випадково, підключають модуль `random`:

```
import random
```

Тоді в програмному коді можна отримати випадкові значення величин різних типів (табл. 25.4).

Таблиця 25.4

Опис	Значення
`random.random()`	Випадкове дійсне число від 0,0 включно до 1,0
`random.randint(1,10)`	Випадкове ціле число від 1 до 10 включно
`random.choice(['а','о','у', 'е','и','і', 'ї' 'ю'])`	Голосна літера українського алфавіту, обрана випадково

У програмах, написаних мовою програмування *Free Pascal*, у програмному коді опрацювання події один раз звертаються до функції `Randomize`, яка підключає генератор випадкових чисел та надає можливість отримувати щоразу інші значення. Тоді в програмному коді можна отримати випадкові значення величин різних типів (табл. 25.5).

Таблиця 25.5

Опис	Значення змінної A
`Randomize;` `A:=random;`	Випадкове дійсне число від 0,0 включно до 1,0
`Randomize;` `A:=random(10);`	Випадкове ціле число від 0 включно до 10
`Randomize;` `A:=random(b-a+1)+a;`	Випадкове ціле число від a до b включно

ДІЄМО

Вправа 4. Гра.

Завдання. Складіть програму мовою програмування *Python*, яка реалізує гру *Відгадай число* між комп'ютером та користувачем. Користувач не більш ніж за 6 спроб має відгадати «задумане» число з діапазону від 1 до 20. Якщо користувач відгадає, то на екран виводиться повідомлення про кількість спроб, якщо ні — виводиться «задумане» число.

1. Відкрийте середовище програмування *PyCharm*.
2. Створіть новий файл програми мовою *Python* з іменем *Відгадай число* в папці *Навчальні проекти* власної структури папок.
3. У програмному коді запишіть команди за зразком. Проаналізуйте, яке призначення кожного рядка програмного коду.

```
import random
guessesTaken = 0
print('Привіт! Як тебе звати?')
myName = input ()
number = random.randint(1, 20)
print('Отже, ' + myName + ', Я задумав число від 1 до 20.')
while guessesTaken < 6:
    print('Спробуй відгадати.')
    guess = input ()
```

```
    guess = int(guess)
    guessesTaken = guessesTaken + 1
    if guess < number:
      print('Твоє число менше від задуманого.')
    if guess > number:
      print('Твоє число більше за задумане.')
    if guess == number:
      break
if guess == number:
    guessesTaken = str(guessesTaken)
    print('Дуже добре, ' + myName + '! Для відгадування
    тобі знадобилося ' + guessesTaken + ' разів!')
if guess != number:
    number = str(number)
    print('Ні. Число, яке я задумав - ' + number)
```

4. Запустіть проект на виконання. Спробуйте виграти у грі. Завершіть роботу із середовищем.

ОБГОВОРЮЄМО

Обговоріть запитання, що містяться у файлі *Тема 25*, який зберігається у папці *Обговорюємо*. Також відкрити ці запитання можна за допомогою QR-коду.

ПРАЦЮЄМО В ПАРАХ

1. Обговоріть, як скласти схему для узагальнення опису циклів із повторенням мовами програмування *Free Pascal* та *Python.* Побудуйте таку схему засобами текстового процесора для обраної мови програмування. Порівняйте побудовані схеми зі схемами, виконаними іншою парою учнів.
2. Обговоріть, чим відрізнятиметься проект *Урожай*, розроблений мовою програмування *Python* від аналогічного, розробленого мовою програмування *Free Pascal*. Реалізуйте такий проект у середовищі *Lazarus*.
3. Обговоріть, чим буде відрізнятися проект *Просте_число*, розроблений мовою програмування *Free Pascal* від його аналогічного, розробленого мовою програмування *Python*. Реалізуйте такий проект у середовищі *PyCharm*.

ПРАЦЮЄМО САМОСТІЙНО

1. Яких значень буде надано змінним, імена яких трапляються в командах, у результаті виконання таких фрагментів програм (для різних значень вихідних даних), поданих у таблиці 2.6.

Мова програмування	Опис фрагмента програми	Вихідні дані

Free Pascal	1)
	`I:=N;`
	`while I<5 do`
	`  begin`
	`  X:=X+1;`
	`  I:=I+1`
	`  end;`
		при 1) N=6; X=0
		2) N=2; X=0
		3) N=4; X=0

	2)
	`for I:=1 to 5 do`
	`  begin`
	`  N:=N+2;`
	`  S:=S+N`
	`  end;`
		при 1) N=1; S=0
		2) N=2; S=0
		3) N=4; S=-1

Python	3)
	`I:=1; X:=0;`
	`while (I< N) and (Y='ТАК'):`
	`  X:=X+1;`
	`  I:=I+1`
		при 1) N=3; Y='ТАК'
		2) N=3; Y='НЕМАЄ'
		3) N=1; Y='ТАК'
		4) N=1; Y='НЕМАЄ'
		5) N=5; Y='ТАК'

	4)
	`for i in range(1, 5):`
	`  for J in range(1, 5):`
	`  r= i*J`
	`  print(I+ ', що множиться на '+J+ ' дорівнює'+r);`	

2. Запишіть фрагменти програм мовами програмування *Free Pascal* та *Python* для виконання таких дій:

1) друкується таблиця множення на 9;
2) друкуються квадрати чисел від 10 до 1;
3) знаходиться сума цілих чисел від 1 до 10;
4) обчислюється добуток парних чисел першого десятка;
5) обчислюється сума дробів, знаменниками яких є непарні числа від 3 до 9;
6) визначається кількість введених речень (речення закінчується крапкою, знаком оклику чи знаком питання).

3. Розробіть проект у вибраному середовищі програмування, за допомогою якого можна визначити найменший спільний дільник двох цілих чисел.

4. Розробіть проект у вибраному середовищі програмування, у якому за введеним початковим і кінцевим значенням температури та кроком їх зміни в текстове поле буде виведений список відповідних температур за шкалою Фаренгейта, якщо для такого переведення використовують формулу:

$$T_f = \frac{9}{5} T_C + 32.$$

5. Розробіть проект у вибраному середовищі програмування, у якому у введеному виразі, що містить арифметичні операції без дужок і математичних функцій, у повідомленні буде виводитись, які дії потрібно виконати.

6. Розробіть проект у вибраному середовищі програмування, у якому із тексту, який вводиться з клавіатури, буде виводитися та його частина, що розташована: 1) до першої крапки; 2) від другої крапки й до кінця.

26. ПРАКТИЧНА РОБОТА 12

СКЛАДАННЯ ТА ВИКОНАННЯ АЛГОРИТМІВ З ПОВТОРЕННЯМИ Й РОЗГАЛУЖЕННЯМИ ДЛЯ ОПРАЦЮВАННЯ ВЕЛИЧИН

ПРИГАДАЙТЕ

- як описати алгоритмічну структуру розгалуження мовами програмування *Free Pascal* та *Python*;
- як описати алгоритмічну структуру повторення мовами програмування *Free Pascal* та *Python*;
- як описати операції над величинами різних типів мовами програмування *Free Pascal* та *Python*;
- для чого на електронних формах використовують елементи управління *перемикач, прапорець, список*.

СТВОРІТЬ

У власній структурі папок створіть папку *Практична робота 12.*

ПАМ'ЯТАЙТЕ

Під час виконання практичних завдань пам'ятайте про правила безпеки життєдіяльності при роботі з комп'ютером!

Завдання 1. Розрахунок оплати за електроенергію (*16 балів*)

В Україні за споживання електроенергії визначено такий порядок оплати залежно від спожитого обсягу: вартість 1 кВт при споживанні електроенергії до 100 кВт за місяць — 45,6 грн, від 100 кВт до 600 кВт — 78,9 грн, понад 600 кВт — 147,9 грн.

Деяким споживачам надається пільговий обсяг безоплатного користування електроенергією, наприклад, 30 кВт. Якщо така пільга є, то її віднімають від спожитого обсягу, а далі розрахунок здійснюється за прийнятими тарифами.

Розробіть у вибраному середовищі програмування проект *Оплата за електроенергію*, у якому користувач вносить у текстове поле обсяг спожитої електроенергії, та позначає за допомогою елементів управління (визначте самостійно) наявність пільги. Після натиснення кнопки *Розрахувати* отримує у вікні повідомлення розмір суми для оплати за електроенергію.

Завдання 2. Пенсійний калькулятор (*19 балів*)

Мама восьмикласника Максима працює у відділенні Пенсійного фонду. Максим запропонував розробити для неї програму, яку вона може використати у своїй роботі.

В Україні визначено такий порядок оподаткування пенсії з фізичних осіб:

- встановлюється мінімальна заробітна плата (наприклад, із січня по квітень — 1378 грн, із травня по листопад — 1450 грн, із грудня — 1550 грн);
- якщо сума нарахування s менша, ніж три мінімальні заробітні плати, то пенсія не оподатковується;
- якщо сума нарахування s більша за три мінімальні заробітні плати, але не перевищує 10 мінімальних заробітних плат, то із суми, яка перевищує три мінімальні заробітні плати, нараховують 15 % податку;

- якщо сума нарахування s більша, ніж 10 мінімальних заробітних плат, то оподатковується сума, що перевищує три мінімальні зарплати. Із суми, що залишилась до 10 мінімальних заробітних плат, нараховують 15 % податку, а із суми, що перевищує 10 мінімальних заробітних плат, нараховують 20 % податку.

Розробіть у вибраному середовищі програмування проект *Пенсійний калькулятор*, у якому користувач вносить у текстове поле розмір нарахованої пенсії та обирає за допомогою елементів управління (визначте самостійно) розмір мінімальної заробітної плати. Після натиснення кнопки *Розрахувати* в текстових полях, захищених від змін, отримує розміри суми нарахованої пенсії та податку.

Завдання 3. Шалені знижки (*16 балів*)

У магазині на деякі товари (молоко, фрукти, овочі, печиво) прийняли систему знижок: товар, який не проданий за тиждень, наступного тижня дешевшає на 10 %, ще через тиждень — на 20 % і т. д.

Розробіть у вибраному середовищі програмування проект *Знижки*, у якому користувач обирає зі списку товар й у текстове поле вносить номер тижня покупки. Після натиснення кнопки *Розрахувати* у вікні повідомлення — отримує ціну товару на дату покупки. Візьміть до уваги, що відсотки знижки розраховуються, починаючи із другого тижня за формулою:

$$price2 = price1 * \frac{100 - 1 * 10}{100}$$

$$price3 = price2 * \frac{100 - 2 * 10}{100}$$

...

Завдання 4. Кролеферма (*18 балів*)

Ваш дідусь у селі вирішив розводити кроликів. Він знає, що ще в XIII ст. Леонардо Пізанський вивів формулу закону розмноження кроликів, яка визначає ряд чисел, що отримали назву чисел Фібоначчі: 1, 1, 2, 3, 5, 8...

Можна помітити, що, починаючи з третього числа, кожне наступне дорівнює сумі двох попередніх, тобто має місце формула:

$$F_n = F_{n-1} + F_{n-1}.$$

Розробіть проект у визначеному середовищі програмування, за яким після натиснення кнопки *Розрахувати* за введеним у текстове поле порядковим номером, що визначає послідовність етапу розмноження кроликів, у вікні повідомлення буде відображено відповідне число Фібоначчі.

Завдання 5. Порівняння (*12 балів*)

Члени математичного гуртка ознайомились із поняттям факторіала числа — добутку натуральних чисел від 1 до даного числа: $n!=1\cdot2\cdot3\cdot...n$.

Розробіть проект у визначеному середовищі програмування, з допомогою якого можна буде порівняти значення фаторіала чила та його квадрата.

27. ГРАФІЧНЕ ВІДОБРАЖЕННЯ ДАНИХ МОВАМИ ПРОГРАМУВАННЯ

ПРИГАДАЙТЕ:

- які інструменти малювання використовують у середовищі графічного редактора;
- як створюють та змінюють формат малюнків у середовищі текстового процесора;
- чим відрізняються малюнки, створені у графічному редакторі та текстовому процесорі;
- засоби для малювання в навчальному середовищі створення та виконання алгоритмів *Скретч*.

ВИ ДІЗНАЄТЕСЯ:

- як у середовищі програмування *Lazarus* використовують малюнки, збережені в зовнішніх файлах;
- як у середовищі *Lazarus* на форму додають графічні фігури;
- як у середовищі *Lazarus* малюють лінію, сектор та ламану;
- як побудувати зображення в середовищі програмування *PyCharm*.

ВИВЧАЄМО

1. Як у середовищі програмування *Lazarus* використовують малюнки, збережені в зовнішніх файлах?

У програмах часто використовують малюнки: ілюстрації, рухомі зображення, фони тощо. Деякі середовища програмування мають засоби, які забезпечують додавання готових зображень до програмного коду або створення й форматування малюнків у самій програмі. Наприклад, у навчальному середовищі створення та виконання алгоритмів *Скретч* ви використовували такі засоби для роботи з графікою:

- змінювали образи об'єктів, завантажуючи їх із бібліотеки;
- малювали об'єкти у вбудованому графічному редакторі;
- виконували побудову зображень виконавцем, вказуючи у програмному коді команди групи *Олівець*.

Середовище програмування *Lazarus* містить засоби для роботи з графічними зображеннями. На формах можна розміщувати малюнки, які збережені в зовнішніх файлах. Компоненти екранної форми, які використовують для розміщення графічних об'єктів, містяться на панелі компонентів *Additional* (*Додаткова*). Для вставлення контейнера, до якого додають малюнок із файла, на формі розміщують компонент `Image` (мал. 27.1).

Мал. 27.1

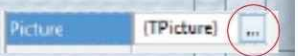
Компонент `Image` має основні властивості, подані в таблиці 27.1.

Таблиця 27.1

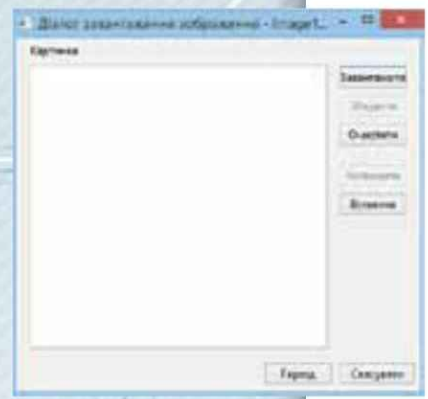
Властивість	Опис
`Picture`	Графічне зображення, яке відображається в полі компонента
`Width, Height`	Розмір компонента. Якщо розмір компонента менший, ніж розмір зображення, і значення властивостей `AutoSize`, `Strech` і `Proportional` дорівнює `False`, то відображається частина зображення
`Proportional`	Автоматичне масштабування картинки зі збереженням пропорцій, без спотворення (значення `True`)
`Strech`	Автоматичне масштабування (стиснення чи розтягнення) зображення відповідно до реального розміру компонента. Якщо розмір компонента не пропорційний розміру зображення, то зображення буде спотворене
`AutoSize`	Автоматична зміна розміру компонента до реального розміру зображення
`Visible`	Відображення зображення на формі
`Canvas`	Поверхня, на яку можна вивести зображення

Для того щоб змінити значення властивості `Picture`, використовують алгоритм додавання зображення на форму:

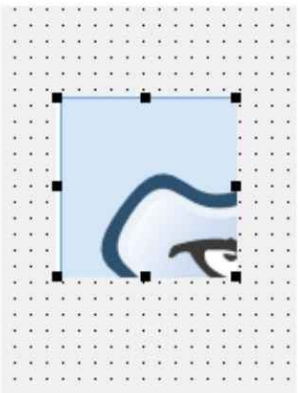
1. Викликати вікно завантаження графічного зображення в рядку цієї властивості в *Інспекторі об'єктів* (мал. 27.2).
2. Натиснути кнопку *Завантажити* у вікні *Діалог завантаження зображення*. У вікні, що відкривається при цьому, обрати потрібний графічний файл і натиснути кнопку *Відкрити*. Далі у вікні перегляду буде відображено вміст цього файла (мал. 27.3).
3. Натиснути кнопку *Гаразд* і перейти до надання значень властивостям компонента (мал. 27.4).



Мал. 27.2



Мал. 27.3

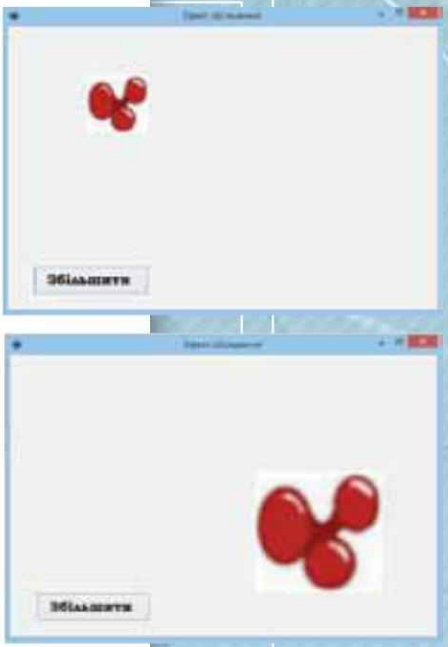


Мал. 27.4

ДІЄМО

Вправа 1. Озон.

Завдання. У середовищі програмування *Lazarus* створіть проект, форма якого містить малюнок, що збережений у файлі *Ozon.jpg* у папці *Програмування*. При натисненні кнопки *Збільшити* розміри малюнка збільшуються (імітується ефект наближення, мал. 27.5).



Мал. 27.5

1. Сплануйте проект. Передбачте, які об'єкти будуть використані на екранній формі та які події будуть відбуватись із ними.
2. У папці *Навчальні проекти* власної структури папок створіть папку *Озон*.
3. Відкрийте середовище *Lazarus*, створіть новий проект. Змініть значення властивостей об'єкта `Form1`, розмістіть на формі об'єкти та надайте значень їх властивостям (табл. 27.2).

Таблиця 27.2

Об'єкт	Властивість	Значення
`Form1`	`Caption`	Ефект збільшення
`Button1`	`Caption`	Збільшити
	`Font`	`Bookman Old Style`, жирний, 16
`Image1`	`Picture`	*Ozon.jpg*
	`Strech`	True

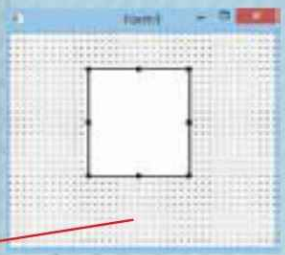
4. Створіть процедуру опрацювання події *натиснення кнопки Збільшити*. У редакторі коду введіть команди для переміщення малюнка та збільшення його:

```
Image1.Top := Image1.Top + 100;
Image1.Left  := Image1.Left+250;
Image1. Height  := Image1.Height + 100;
Image1.Width  :=Image1.Width + 100;
```

5. Запустіть проект на виконання. Перевірте, чи отримали ви потрібний результат.
6. Завершіть роботу з проектом і середовищем програмування.

2. Як у середовищі *Lazarus* на форму додають графічні фігури?

У середовищі *Lazarus* можна розміщувати деякі геометричні фігури на формі за допомогою компонентів або створити програмний код для малювання фігур у процесі виконання програми. Для розміщення на формі фігур, а саме: прямокутника, еліпса, трикутника, ромба тощо — використовують компонент *Shape* панелі компонентів *Aditional* (*Додаткова*). За замовчуванням на екранній формі буде розміщено прямокутник (мал. 27.6).



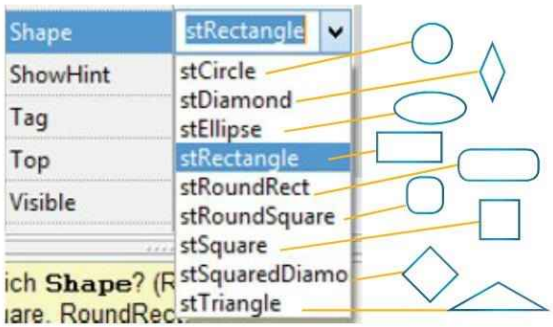
Мал. 27.6

Щоб змінити форму фігури, використовують властивість `Shape` (мал. 27.7).

Щоб змінити колір фігури та стиль заливки, використовують властивість `Brush`.

Наприклад, `Shape1.Brush.Color:=clRed` — колір об'єкта `Shape1` буде червоним, `Shape1.Brush.Style:=bsSolid` — суцільне зафарбовування.

У середовищі *Lazarus* властивість `Color` може набувати фіксованого набору значень (табл. 27.3).



Мал. 27.7

Таблиця 27.3

Значення	Колір	Значення	Колір	Значення	Колір
clBlack	Чорний	clSilve	Срібний	clPurple	Рожевий
clMaroon	Каштановий	clRed	Червоний	clTeal	Пурпурний
clGreen	Зелений	clLime	Салатовий	clGray	Сірий
clOlive	Оливковий	clBlue	Синій	clAqua	Бірюзовий
clNavy	Темно-синій	clFuchsia	Яскраво рожевий	clWhite	Білий

Властивість `Style` може набувати значень із набору, вказаного в таблиці 27.4.

Таблиця 27.4

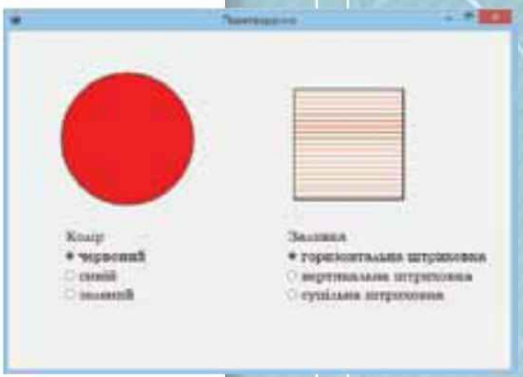
Значення	Пояснення	Значення	Вигляд
bsSolid	Суцільне зафарбування	bsHorizontal	Горизонтальна штриховка
bsClear	Не зафарбовується	bsVertical	Вертикальна штриховка
bsDFiagonal	Діагональна штриховка з нахилом ліній вперед	bsBFiagonal	Діагональна штриховка з нахилом ліній назад
bsCross	Горизонтально-вертикальна штриховка в клітинку	bsDiagCross	Діагональна штриховка в клітинку

До об'єктів `Shape` найчастіше застосовують події переміщення миші `OnMouseMove`, натиснення та відпускання кнопки миші `OnMouseDown` та `OnMouseUp`. Кожна з двох останніх подій відбувається, якщо клацнути мишею на фігурі, яка розміщена на формі, — у цьому разі відбувається і натиснення, і відпускання кнопки миші.

ДІЄМО

Вправа 2. Перетворення.

Завдання. Розробіть проект у середовищі *Lazarus*, у якому, коли клацають мишею на зображенні круга, він зафарбовується в колір, обраний у групі перемикачів *Колір*, а коли клацають мишею зображення квадрата, — зафарбовується обраним кольором та застосовується стиль заливки, який обирають у групі *Заливка* (мал. 27.8).



Мал. 27.8

1. Сплануйте проект. Передбачте, які об'єкти будуть використані на екранній формі та які події будуть відбуватись із ними.
2. У папці *Навчальні проекти* власної структури папок створіть папку *Перетворення*.
3. Відкрийте середовище *Lazarus*, створіть новий проект. Змініть властивості об'єкта `Form1`, розмістіть на формі об'єкти та надайте значень їх властивостям (табл. 27.5).

Таблиця 27.5

Об'єкт	Властивість	Значення
`Form1`	`Caption`	Перетворення
`Shape1`	`Shape`	stCircle
`Shape2`	`Shape`	stSquare
`radioGroup1`	`Items`	Червоний, синій, зелений
`radioGroup2`	`Items`	Горизонтальна штриховка, вертикальна штриховка, суцільна штриховка

4. Створіть процедуру опрацювання події *клацання кнопкою миші на колі*. Для цього оберіть у таблиці вікна *Інспектора об'єктів* вкладку *Події*, двічі клацніть у клітинці праворуч від `OnMouseDown`. У вікні редактора коду запишіть програмний код (мал. 27.9).

```
procedure TForm1.Shape1MouseDown(Seder: TObject; Button: TMouseButton;
    Shift: TShiftState; X, Y: integer);
begin
    if radioGroup1.ItemIndex=0 then Shape1.Brush.Color:=clRed;
    if radioGroup1.ItemIndex=1 then Shape1.Brush.Color:=clBlue;
    if radioGroup1.ItemIndex=2 then Shape1.Brush.Color:=clGreen;
end;
```

Мал. 27.9

5. Створіть процедуру опрацювання події *клацання кнопкою миші на квадраті*.
У вікні редактора коду запишіть програмний код (мал. 27.10).

```
procedure TForm1.Shape2MouseDown(Seder: TObject; Button: TMouseButton;
    Shift: TShiftState; X, Y: integer);
begin
     if radioGroup1.ItemIndex=0 then Shape2.Brush.Color:=clRed;
     if radioGroup1.ItemIndex=1 then Shape2.Brush.Color:=clBlue;
     if radioGroup1.ItemIndex=2 then Shape2.Brush.Color:=clGreen;
     if radioGroup2.ItemIndex=0 then Shape2.Brush.Style:=bsHorizontal;
     if radioGroup2.ItemIndex=1 then Shape2.Brush.Style:=bsVertical;
     if radioGroup2.ItemIndex=2 then Shape2.Brush.Style:=bsSolid;
end;
```

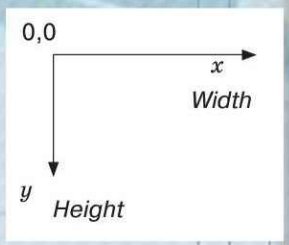
Мал. 27.10

6. Запустіть проект на виконання. Перевірте, чи змінюють свої властивості геометричні фігури за умовою завдання та обраними значеннями властивостей.
7. Завершіть роботу з проектом і середовищем програмування.

3. Як у середовищі *Lazarus* малюють лінію, сектор та ламану?

Для малювання ліній і фігур у процесі виконання програми на формі в середовищі програмування *Lazarus* потрібно розміщувати полотно. Для цього звертаються до властивості `Canvas` — властивості об'єкта `Form`, який у свою чергу є об'єктом, що має свої методи та властивості. Методи використовують для побудови на полотні різних фігур, а властивості — щоб задавати колір лінії, її товщину тощо.

Розмір полотна визначається значеннями властивостей `Height` — кількість точок по вертикалі, та `Width` — по горизонталі. Полотно складається з окремих точок — пікселів, координати яких задаються значеннями x та y (мал. 27.11).



Мал. 27.11

Як і в графічному редакторі, у середовищі програмування *Lazarus* можна малювати олівцем та пензликом. Для цього використовують властивості полотна `Pen` та `Brush`:

`Form1.Canvas.Pen` або `Form1.Canvas.Brush`

Можна задавати значення властивостей інструментів малювання:

`Form1.Canvas.Pen.Color` — колір (див. табл. 27.3);

`Form1.Canvas.Pen.Width` — ширина (ціле число);

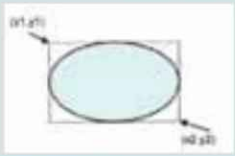
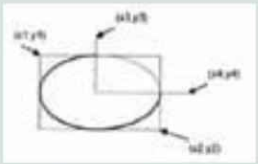
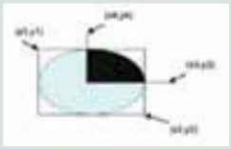
`Form1.Canvas.Pen.Style` — стиль (табл. 27.6).

Таблиця 27.6

Константа	Вигляд	Константа	Вигляд
psSolid	———	psDot	- - - - - - - - -
psDash	— — —	psDashDot	—·—·—·—·—·—
psDashDotDot	··—··—··—··	psClear	Не відображається

Малювати можна різні об'єкти: лінію, ламану, еліпс, сектор. Для цього використовують відповідні методи полотна (табл. 27.7). Колір, товщина та стиль цих об'єктів визначаються значеннями властивості `Pen`, а колір і стиль заливки замкнених фігур — значеннями властивості `Brush`.

Таблиця 27.7

Опис	Пояснення
`Form1.Canvas.LineTo(x,y)`	Побудова лінії до точки з координатами x, y
`Form1.Canvas.MoveTo(x,y)`	Переміщення олівця у точку з координатами x, y
`Form1.Canvas.Restangle(x1,y1,x2,y2)`	Прямокутник із верхнім лівим кутом $x1$,$y1$ та нижнім правим $x2$,$y2$
`Form1.Canvas.` `Polyline(x1,y1,x2,y2,x3,y3)`	Малювання ламаної з координатами $x1$,$y1$,$x2$,$y2$,$x3$,$y3$
`Form1.Canvas.Ellipse(x1,y1,x2,y2)`	Побудова еліпса 
`Form1.Canvas.` `Arc(x1,y1,x2,y2,x3,y3,x4,y4)`	Побудова дуги 
`Form1.Canvas.` `Pie(x1,y1,x2,y2,x3,y3,x4,y4)`	Побудова сектора 

ДІЄМО

Вправа 3. Малюнок із ліній.

Завдання. У середовищі програмування *Lazarus* розробіть проект *Лінії*, у якому на формі малюватимуться лінії за допомогою протягування мишею: початок лінії буде розташовано в позиції, в якій натиснули кнопку миші, а кінець — де відпустили кнопку миші.

1. Сплануйте проект. Передбачте, які події будуть відбуватись на екранній формі та якими засобами мови програмування *Free Pascal* їх можна реалізувати.
2. У папці *Навчальні проекти* власної структури папок створіть папку *Лінії*.
3. Відкрийте середовище *Lazarus*, створіть новий проект. Змініть властивість `Caption` об'єкта `Form1` на *Малювання ліній*.
4. Створіть процедуру опрацювання події *клацнули кнопкою миші*. Для цього оберіть у таблиці вікна *Інспектора об'єктів* вкладку *Події*, двічі клацніть у клітинці праворуч від `OnMouseDown`. У вікні редактора коду запишіть програмний код для оголошення змінної логічного типу `DownM`, яка передаватиме стан натиснення миші в кожну процедуру опрацювання події (мал. 27.12). Зверніть увагу, що ми записуємо її у програмний код до опису процедури опрацювання події!

Цікаво

Змінну у програмі, значення якої можуть бути використані та змінені в усіх процедурах, що складають програму, називають **глобальною змінною**.

```
var
   Form1: TForm1;
   DownM: Boolean;
implementation
```

Мал. 27.12

5. У програмний код запишіть процедуру опрацювання подій для форми `OnMouseDown` — натиснута кнопка миші, `OnMouseUp` — відпущена кнопка миші, `OnMouseMove` — переміщення миші (мал. 27.13).

```
procedure TForm1.FormMouseDown(Seder: TObject; Button: TMouseButton;
  Shift: TShiftState; X, Y: integer);
begin
  DownM := True;
  Canvas.MoveTo(X,Y);

end;

procedure TForm1.FormMouseMove(Seder: TObject; Shift: TShiftState; X, Y: integer);
begin
  if DownM = True then Canvas.LineTo (X,Y);
end;

procedure TForm1.FormMouseUp(Seder: TObject; Button: TMouseButton;
  Shift: TShiftState; X, Y: integer);
begin
  DownM := False;
end;
```

Мал. 27.13

6. Запустіть проект на виконання. Спробуйте намалювати лініями своє ім'я.
7. Завершіть роботу з проектом і середовищем програмування.

4. Як побудувати зображення в середовищі програмування *PyCharm*?

У середовищі програмування *PyCharm* побудова ліній та інших графічних об'єктів у процесі виконання програми подібна до описаного середовищі *Lazarus*.

Для побудови графічних об'єктів викликають відповідний метод модуля `tkinter`, який імпортують до проекту (мал. 27.14).

```
tkinter.Canvas(<назва форми>, width=<значення>, height=<значення>)
```

Створення полотна розміром, який заданий значеннями `width` — ширина, `height` — висота

Мал. 27.14

Створення зображень на полотні викликається методом:

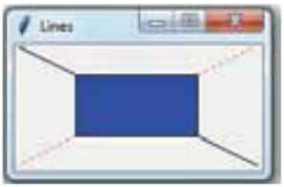
`<Ім'я_об'єкта_полотно>. create_ <об'єкт >.`

Можна побудувати такі об'єкти:

- **лінія** — `line(x1,y1,x2,y2)`
- **прямокутник** — `rectangle(x1,y1,x2,y2)`
- **ламана** — `polygon(x1,y1,x2,y2,x3,y3)`
- **еліпс** — `oval(x1,y1,x2,y2)`
- **дуга** — `arc(x1,y1,x2,y2,x3,y3,x4,y4)`

Об'єкти можуть мати параметри: fill — колір заливки, dash — тип заливки.

Наприклад, для малювання графічних примітивів (мал. 27.15) у редакторі коду записують програмний код (мал. 27.16).



Мал. 27.15

```
import tkinter
main = tkinter.Tk()
main.wm_title("Lines")
# створення полотна для малювання та розміщення
на головній формі
cnv = tkinter.Canvas(main, width=200, height=100)
cnv.pack()
# команди малювання - створення графічних примітивів
на полотні
cnv.create_line(0, 0, 200, 100)
cnv.create_line(0, 100, 200, 0, fill="red", dash=(4, 4))
cnv.create_rectangle(50, 25, 150, 75, fill="blue")
# запуск опрацювання подій програми
main.mainloop()
```

Мал. 27.16

ДІЄМО

Вправа 4. Стовпчаста діаграма.

Завдання. У середовищі програмування *PyCharm* розробіть проект, у якому у вікні головної форми буде побудована стовпчаста діаграма, що демонструє співвідношення між числами 15, 50, 70, 25, 10, 30.

1. Відкрийте середовище *PyCharm*. У папці *Навчальні проекти* власної структури папок створіть проект *Діаграма*, у якому створіть нову програму мовою *Python* з іменем *Побудова_діаграми*.
2. У вікні програмного коду запишіть програму (мал. 27.17).

```
import tkinter
data = [15, 50, 70, 25, 10, 30]
main = tkinter.Tk()
main.wm_title("Data")
# створення полотна для малювання та розміщення
на головній формі
cnv = tkinter.Canvas(main, width=200, height=100)
cnv.pack()
# команди малювання -
створення графічних примітивів на полотні
cnv.create_line(0, 100, 200, 100)
i = 0
while i < len(data):
 cnv.create_rectangle(10 + i * 30, 100, 30 + i * 30,
100 - data[i], fill="blue")
 i += 1
# запуск опрацювання подій програми
main.mainloop()
```

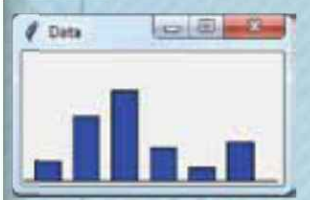
Оголошення списку значень

Введення заголовку форми

Розміри форми

Побудова 6 прямокутників за координатами

Мал. 27.17



Мал. 27.18

3. Запустіть проект на виконання. Перевірте, чи отримали ви вікно форми, зображене на малюнку 27.18.

ОБГОВОРЮЄМО

Обговоріть запитання, що містяться у файлі *Тема 27*, який зберігається у папці *Обговорюємо*. Також відкрити ці запитання можна за допомогою QR-коду.

ПРАЦЮЄМО В ПАРАХ

1. Обговоріть, як скласти таблицю узагальнення щодо роботи із графічними об'єктами в середовищі програмування *Lazarus*. Складіть її засобами текстового процесора.
2. Обговоріть, чим подібне та відрізняється малювання фігурами й побудова цих фігур по точках у середовищі програмування *Lazarus*. Засобами текстового процесора побудуйте діаграму Венна.
3. Обговоріть, як зміниться проект *Лінії*, якщо на форму розмістити кнопку, натиснення якої викликає подію заливки прямокутника кольором форми:

```
form1.canvas.brush.Color:=clwhite;
form1.canvas.FillRect(0,0,500,300);
```

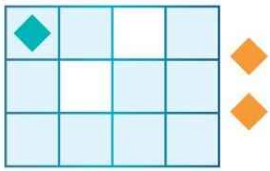
Які елементи управління потрібно використати, щоб у проекті можна було також задавати розмір та колір лінії малювання? Доопрацюйте проект за обговореними ідеями.
4. Обговоріть ідею проекту, у якому можна продемонструвати різні стилі заливки фігур. Реалізуйте цей проект у середовищі програмування *Lazarus*. Розподіліть ролі: один розробляє графічний інтерфейс, інший — складає програмний код.
5. Обговоріть ідею проекту, у якому можна продемонструвати побудову ліній, прямокутників, еліпсів, ламаних, дуг та секторів. Реалізуйте цей проект у середовищах програмування *Lazarus* і *PyCharm*. Розподіліть ролі: один складає проект у середовищі *Lazarus,* інший — у *PyCharm*.

ПРАЦЮЄМО САМОСТІЙНО

1. Створіть проект *Полюси магнітів*, який містить два малюнки магнітів, розташовані один навпроти одного. Потрібні зображення збережені в папці *Програмування* у файлах *Магніт1*, *Магніт 2*. Коли клацнути мишею на лівому малюнку, вони обидва наближаються один до одного, а на правому — віддаляються.
2. Створіть проект *Кошеня*, у якому кошеня після натиснення кнопки *Бігти* змінює своє положення. Різні вигляди кошеняти отримайте із зображення *Кошеня1*, яке збережене в папці *Програмування* відомими вам засобами опрацювання зображень (мал. 27.19).

Мал. 27.19

3. Створіть проект *Площа фігур*, вікно якого містить зображення трикутника, прямокутника й кола, та опрацьовуються такі події: при наведенні вказівника миші на кожну фігуру та утриманні натиснутою її лівої кнопки на екранній формі в написі виводиться формула площі відповідної фігури; якщо кнопку відпустити, формула зникає.
4. Розробіть проект *Прямокутник*, у якому в текстові поля чи у список даних вводять пари координат чисел, натискають кнопку *Побудувати* й отримують прямокутник, побудований за вказаними значеннями координат.
5. Розробіть проект *Природні зони*, у якому на формі розміщене зображення карти України, збережене у файлі *Карта* в папці *Програмування*. Після вибору у списку однієї із природних зон рівнинної частини України: зона мішаних і широколистяних лісів, зона лісостепу, зона степу — межі відповідної зони виділяються ламаною червоного кольору.
6. Розробіть проект *Енциклопедія*, у якому на екранній формі *Сенсорна система* після натиснення кнопки *Відобразити* можна отримати відомості про основні сенсорні системи: зорову, слухову, смаку, нюху, рівноваги, руху, дотику, температури, болю. Спосіб відображення відомостей — малюнок, текст обирається за допомогою елементів управління. Вибір сенсорної системи — відповідним перемикачем. Потрібні відомості та зображення знайдіть в Інтернеті.
7. Розробіть проект *Пелюстки*, у якому обирають розмір пелюстки для малювання за допомогою перемикача, а після натиснення кнопки *Малювати* на екрані будується зображення з 5 пелюсток — секторів, координати яких задаються випадковим чином із діапазону $150 < x < 350$, $50 < y < 250$.
8. Розробіть гру *Знайди пару*, у якій у таблиці «заховано» 6 пар ромбів, кожна з пар — однакового кольору. При виборі користувачем певної клітинки таблиці з'являється відповідний ромб. Якщо при наступному натисненні з'являється ромб того самого кольору, то обидві фігури «вилітають» із таблиці у праве поле. Якщо ні — вони знову «ховаються» в таблицю (мал. 27.20).



Мал. 27.20

ЦІКАВІ ДЖЕРЕЛА

Курси *Python* для початківців:
http://foxford.ru/courses/199?ref=p191_pythonworldru
Сайт *Python 3* для початківців:
http://pythonworld.ru/
Навчання мові *Python* в ігровій формі:
https://checkio.org/
Курс по бібліотеці *Tkinter* мови *Python*:
https://ru.wikiversity.org/wiki/Курс_по_библиотеке_Tkinter_языка_Python

28. ПРАКТИЧНА РОБОТА 13

СКЛАДАННЯ ТА ВИКОНАННЯ АЛГОРИТМІВ ІЗ ГРАФІЧНИМ ВІДОБРАЖЕННЯМ ДАНИХ

ПРИГАДАЙТЕ

- як у середовищі програмування *Lazarus* використовують малюнки, збережені на зовнішніх носіях, та створюють зображення;
- як будують зображення в середовищі програмування *PyCharm*.

СТВОРІТЬ

У власній структурі папок створіть папку *Практична робота 13.*

ПАМ'ЯТАЙТЕ

Під час виконання практичних завдань пам'ятайте про правила безпеки життєдіяльності при роботі з комп'ютером!

Завдання 1. Рух (*24 бали*)

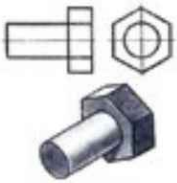
У середовищі програмування *Lazarus* розробіть проект *Рух*, у якому після натиснення кнопки *Поїхати* запускається демонстрація руху автомобілів через перехрестя без поворотів, що регулюється світлофором. Світло світлофора обирається у групі перемикачів *Світлофор*, вигляд автомобіля — у групі прапорців *Транспорт*, допустима швидкість на ділянці дороги — у групі перемикачів *Швидкість* (50, 70, 90 км/год). Світлофор і перехрестя намалюйте засобами середовища програмування, а зображення авто завантажте із файлів *Авто1*, *Авто2*, *Авто3*, які збережені в папці *Програмування.*

Завдання 2. Графічний редактор (*24 бали*)

У середовищі програмування *Lazarus* розробіть проект *Графічний редактор*, у якому можна отримувати зображення штампами — кіл, ромбів, овалів, прямокутників, квадратів, прямокутників із заокругленими краями, повернутих ромбів і квадратів, що обираються за допомогою прапорців та будуються після натиснення кнопки *Побудувати* за початковими координатами, які вводять у текстові поля. Кількість зображень, що повторюються штампом по горизонталі через деякий інтервал, задається у програмі випадково (кількість від 1 до 5, інтервал — від 10 до 50). Колір малювання та стиль зафарбовування обирається у групах відповідних перемикачів.

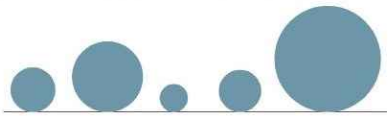
Завдання 3. Креслення деталі (*20 балів*)

Розробіть у середовищі *PyCharm* проект, за допомогою якого на екрані будується креслення деталі. Самостійно доберіть координати геометричних фігур, що складають зображення проекцій, дотримуючись пропорцій на малюнку.



Завдання 4. Бульбашкова діаграма (*22 бали*)

Розробіть проект у середовищі *PyCharm*, за допомогою якого за введеними даними — списком п'яти числових значень — будується бульбашкова діаграма, що відображає «бульбашки», розмір яких відповідає числовим значенням.



РОЗДІЛ 7

ТЕХНОЛОГІЇ ОПРАЦЮВАННЯ ЧИСЛОВИХ ДАНИХ У СЕРЕДОВИЩІ ТАБЛИЧНОГО ПРОЦЕСОРА

29. ОБЧИСЛЕННЯ В ЕЛЕКТРОННИХ ТАБЛИЦЯХ

ПРИГАДАЙТЕ:

- з якими основними об'єктами можна працювати в середовищі табличного процесора;
- як застосовувати засіб автозаповнення для введення даних;
- як копіювати та переміщувати дані з клітинок та діапазонів клітинок;
- як змінювати форматування таблиці;
- як виконувати обчислення з числовими даними таблиці;
- як здійснювати обчислення за допомогою вбудованих функцій;
- що відбувається при копіюванні формул.

ВИ ДІЗНАЄТЕСЯ:

- чим відрізняються абсолютні та відносні посилання на клітинки;
- як додати до формули вбудовану функцію;
- які математичні та статистичні функції найчастіше використовуються в середовищі табличного процесора;
- як у формулах використовують логічні функції;
- як скористатися засобом умовного форматування даних.

ВИВЧАЄМО

1. Чим відрізняються абсолютні та відносні посилання на клітинки?

Ви вже знаєте, що обчислення в електронних таблицях виконують за допомогою формул. У формулах використовують посилання на клітинки або діапазони клітинок, вказуючи їх адреси або ім'я клітинки чи діапазону, що надані користувачем.

Під час роботи з електронними таблицями часто однотипні розрахунки необхідно виконувати в кількох суміжних клітинках. У середовищі табличного процесора, зокрема *Microsoft Excel 2010* або *LibreOffice Calc*, можна в цьому разі не вводити формули кілька разів із клавіатури, а скопіювати формулу з однієї клітинки в іншу за допомогою буфера обміну або скористатися засобом автозаповнення. При цьому також відбувається копіювання формули.

На відміну від копіювання текстових значень, при копіюванні формули, що містить посилання на клітинки, можна отримати формули, які не дублюють вихідну. Це залежить від вигляду посилань на клітинки, які можуть бути **відносними**, **абсолютними** або **комбінованими (мішаними)**. Вигляд посилань має значення лише при копіюванні формули, при обчисленні значення за формулою в одній клітинці вигляд посилань не є суттєвим.

Ви вже створювали формули, які містять відносні посилання — такі посилання використовуються за замовчуванням. При копіюванні формули, що містить **відносні посилання**, такі посилання будуть коригуватися залежно від напрямку копіювання. Відносне посилання на клітинку складається лише з назви стовпця й номера рядка.

Наприклад, якщо клітинка *F4* містить формулу *=D4*E4*, то при копіюванні її до клітинки *F5* вона набуде вигляду *=D5*E5*. Така формула з від-

носними посиланнями при копіюванні в нову клітинку інтерпретується так: знайти добуток значень двох сусідніх зліва клітинок, що в тому самому рядку, що і клітинка, яка містить формулу.

Абсолютне посилання у формулі свідчить, що при копіюванні формули необхідно залишати посилання саме на ту клітинку, адреса якої зазначена, така адреса залишиться незмінною. Щоб відрізнити абсолютне посилання від відносного, у його записі перед назвою стовпця й номером рядка ставиться символ *$*, наприклад, *$C$1* (мал. 29.1).

	A	B	C	D	E
1	Відсоток знижки		5%		
2					
3	**Найменування товару**	**Вартість**	**До сплати**		
4	Навушники A4tech MK-610	109,3	103,84	=B4*(1-C1)	
5	Миша TRUST GXT 25 Gaming Mouse	299	284,05	=B5*(1-C1)	
6	Серветки ColorWay CW-1071	49	46,55	=B6*(1-C1)	
7					

Мал. 29.1

У комбінованому, або мішаному посиланні, назва стовпця є абсолютною, а номер рядка — відносним, або навпаки. Наприклад, *B$4*, *$B4* (мал. 29.2). Коригується при копіюванні лише відносна частина адреси.

	A	B	C	D	E	F	G	H	I	J
1	**Стрибки в довжину**									
2	Спортсмен	Рекорд	Спроба			Відхилення			=C4-$B4	
3			1	2	3	1	2	3		
4	Андрієвський	466	460	468	463	-6	2	-3		
5	Васильковець	485	482	485	480	-3	0	-5		
6						=C5-$B5			=E5-$B5	
7										

Мал. 29.2

Алгоритм виконання обчислень у середовищі табличного процесора з однотипними розрахунками можна подати словесно:

1. Внести в клітинки таблиці дані, які будуть необхідні для виконання обчислень.
2. Визначити, які клітинки в таблиці мають містити однотипні розрахунки, які посилання слід використати для обчислень — абсолютні, відносні чи комбіновані.
3. Створити формулу з посиланнями на клітинки, використовуючи виділення в таблиці потрібних клітинок або діапазонів.
4. Вказати за потреби ті посилання, які є абсолютними чи комбінованими, додавши знак *$* перед назвою стовпця та (або) номером рядка.
5. Скопіювати створену формулу до діапазону клітинок, які мають містити однотипні розрахунки.

Цікаво

У табличному процесорі *Microsoft Excel 2010* можна швидко змінити вигляд посилання у формулі, для цього слід натиснути на клавіатурі клавішу *F4* — її послідовне натиснення змінює відносне посилання на абсолютне, потім на комбіноване з абсолютним номером рядка, далі — на комбіноване з абсолютною назвою стовпця, і знову на відносне посилання.

ДІЄМО

Вправа 1. Розрахунки.

Завдання. Обчисліть вартість проданого товару за допомогою введення формул для даних таблиці, що містяться у файлі *Розрахунки*.

1. Створіть у власній структурі папок папку *Табличний процесор*.
2. Відкрийте файл *Розрахунки*, що зберігається в папці *Табличний процесор*.
3. Виділіть клітинку *D3*, введіть до неї формулу *=C3*C12* (мал. 29.3) та натисніть клавішу *Enter*.
   Ціна в гривнях розраховується як добуток ціни в доларах на курс долара, однак адресу клітинки *C12*, у якій введено курс долара, не потрібно змінювати при копіюванні формули в інші клітинки. Тому формула містить відносну адресу *C3*, яка коригуватиметься при копіюванні, та абсолютну адресу *C12*.
4. Виділіть клітинку *D3* та виконайте протягування за маркер автозаповнення вниз до клітинки *D7*.
5. Для обчислення ціни товару з ПДВ (податок на додану вартість становить

D3 : =C3*C12

	A	B	C	D	E	F	G
1		Розрахунки вартості проданого товару					
2	№	Товар	Ціна в дол.	Ціна в грн.	Ціна з ПДВ	Кількість	Вартість
3	1	Фоторамка	$6,00	141,24 грн.		3	
4	2	Портативна колонка	$10,60			2	
5	3	Електронна книга	$65,00			2	
6	4	Карта пам'яті 16 ГБ	$5,56			5	
7	5	Карта пам'яті 32 ГБ	$12,24			4	
8							
9							
10		Дата виписування рахунку	12.01.16				
11		Дата оплати рахунку	17.01.16				
12		Курс дол.	23,54				
13							
14		Вартість покупки					

Мал. 29.3

20 % від вартості товару) необхідно до ціни товару додати ще 20 % його вартості, тому до клітинки *E3* введіть формулу *=D3+D3*0,2* та натисніть клавішу *Enter*.

Оскільки необхідно, щоб при подальшому копіюванні цієї формули до клітинок стовпця *E* адреса клітинки *D3* змінювалась відповідно на *D4*, потім на *D5*, далі на *D6* та *D7*, слід у формулі лишити відносну адресу клітинки *D3*.

6. За допомогою автозаповнення скопіюйте формулу з клітинки *E3* до клітинок діапазону *E4:E7*.
7. Виділіть клітинку *G3* та введіть формулу *=E3*F3* для обчислення вартості вказаної кількості товару. Скопіюйте цю формулу до діапазону клітинок *G4:G7*.
8. Виділіть клітинку *G14*. За допомогою автосуми знайдіть суму значень діапазону клітинок *G3:G7*.
9. Збережіть результати у файлі з тим самим іменем у папці *Електронні таблиці* власної структури папок.

Вправа 2. Таблиця квадратів.

Завдання. Створіть таблицю квадратів двозначних чисел із використанням формул, що містять комбіновані посилання.

1. Відкрийте файл *Таблиця квадратів*, що зберігається в папці *Електронні таблиці*.

	A	B	C	D	E	F	G	H	I	J	K
1		Таблиця квадратів двозначних чисел									
2		0	1	2	3	4	5	6	7	8	9
3	1	100	121	144	169	196	225	256	289	324	361
4	2	400	441	484	529	576	625	676	729	784	841
5	3	900	961	1024	1089	1156	1225	1296	1369	1444	1521
6	4	1600	1681	1764	1849	1936	2025	2116	2209	2304	2401
7	5	2500	2601	2704	2809	2916	3025	3136	3249	3364	3481
8	6	3600	3721	3844	3969	4096	4225	4356	4489	4624	4761
9	7	4900	5041	5184	5329	5476	5625	5776	5929	6084	6241
10	8	6400	6561	6724	6889	7056	7225	7396	7569	7744	7921
11	9	8100	8281	8464	8649	8836	9025	9216	9409	9604	9801
12											

Мал. 29.4

2. Виділіть клітинку *B3* та введіть до неї формулу *=($A3*10+B$2)^2* (мал. 29.4).
3. Виділіть клітинку з формулою та виконайте протягування за маркер автозаповнення вправо до клітинки *K3*. Не знімаючи виділення з діапазону, виконайте протягування за маркер автозаповнення вниз до клітинки *K11*.
4. Поясніть формулу, що застосована для цієї таблиці.
5. Збережіть результати у файлі з тим самим іменем у папці *Табличний процесор* власної структури папок.

2. Як додати до формули вбудовану функцію?

Ви вже вмієте використовувати для обчислень деякі вбудовані функції табличного процесора для знаходження суми значень діапазону клітинок, середнього значення, максимального чи мінімального значень.

Табличний процесор містить великий набір вбудованих функцій, які можна використати для обчислень та опрацювання даних. Кожна функція має своє ім'я, більшість функцій містить принаймні один необхідний для обчислення значення функції аргумент. Аргументи записуються в круглих дужках і відокремлюються один від одного крапкою з комою (;). Аргументом може бути число, текст, що записаний у лапках, посилання на клітинку чи діапазон клітинок, або вираз, що може також містити функції. Деякі аргументи є обов'язковими, деякі — ні.

Функцію можна ввести з клавіатури, як і будь-який вміст клітинки. Для спрощення введення функцій до формули можна скористатися відповідними інструментами табличного процесора. Для полегшення пошуку всі функції об'єднані в категорії: математичні, статистичні, логічні, фінансові, текстові тощо. Також виділена окрема категорія *Нещодавно використані* в *Microsoft Excel 2010* (*Востаннє вжито* в *LibreOffice Calc*), у якій можна переглянути та обрати функції, що використовувались останнім часом під час роботи на конкретному комп'ютері. Як правило, у цій категорії будуть відображені найчастіше вживані функції, тому в ній можна швидко знайти потрібну функцію.

У табличному процесорі *Microsoft Excel 2010* знайти та додати до формули функцію можна за допомогою інструментів із групи *Бібліотека функцій* на вкладці *Формули* (мал. 29.5).

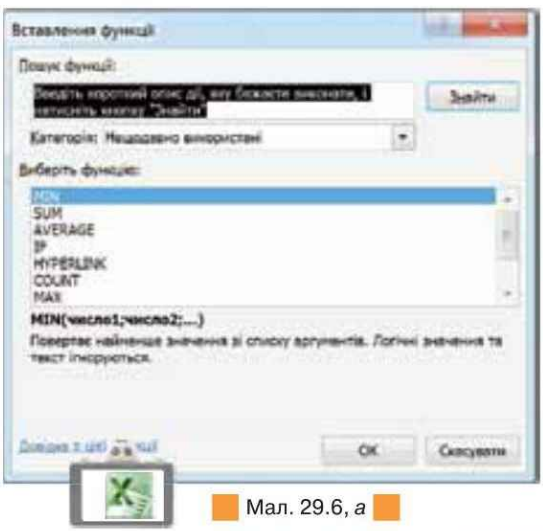
У табличних процесорах ліворуч від рядка формул розташований інструмент *Вставити функцію* — або , за допомогою якого відкривається вікно *Вставлення функції* в *Microsoft Excel 2010* (мал. 29.6, *а*) або *Помічник з функцій* в *LibreOffice Calc* (мал. 29.6, *б*).

Цікаво

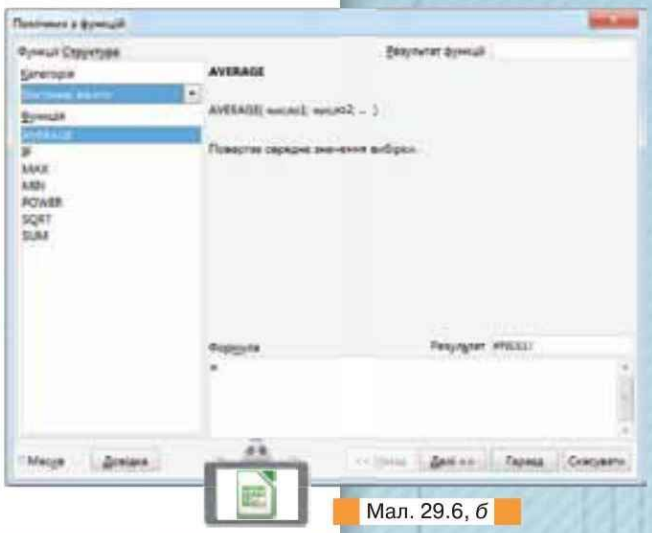
У табличному процесорі *Microsoft Excel* з російським інтерфейсом назви більшості функцій подані скороченнями від російських слів.

Деякі функції не мають аргументів, але навіть у такому випадку круглі дужки все одно мають бути записані. Наприклад, функція PI() повертає значення числа π з точністю до 15 знаків.

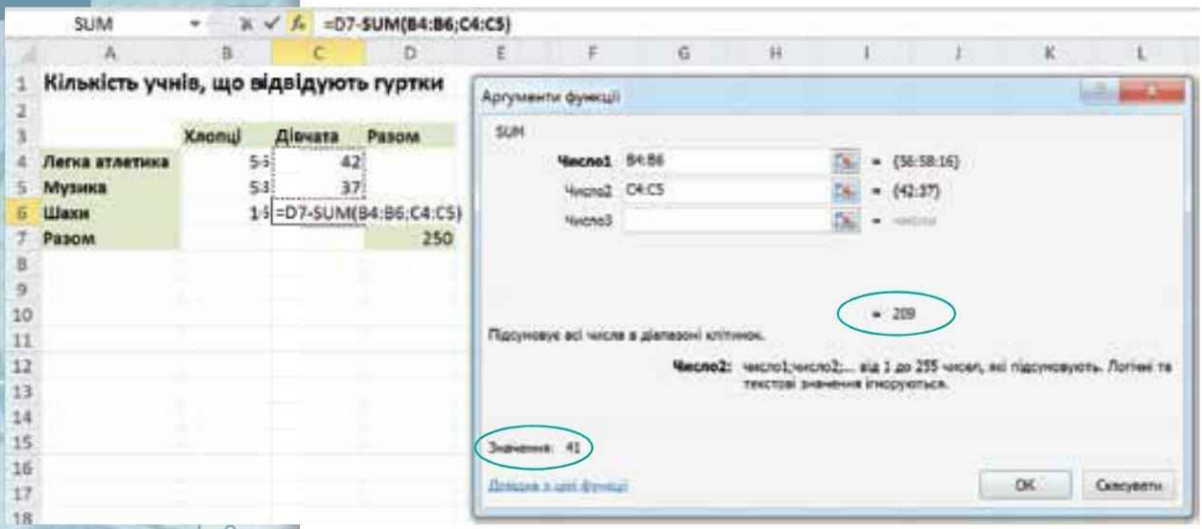
Мал. 29.5



Мал. 29.6, *а*



Мал. 29.6, *б*



Мал. 29.7

У цьому вікні слід спочатку обрати категорію функції та ім'я потрібної функції зі списку доступних функцій обраної категорії. За потреби можна скористатися довідкою про використання кожної функції. На наступному кроці слід задати значення аргументів функції. Це можна зробити, виділивши клітинки на робочому столі за допомогою миші або ввівши адреси клітинок за допомогою клавіатури. Аргументи функцій, як і посилання у формулах, можуть містити абсолютні, відносні або комбіновані посилання — залежно від того, чи має посилання змінюватися під час копіювання формули з функцією. Після завершення введення аргументів буде відображено значення функції, а в області *Значення* — результат обчислення формули (мал. 29.7).

ДІЄМО

Вправа 3. Створення формули з використанням функції для обчислення. Завдання. Учням 8 та 9 класів запропонували обрати один із гуртків: легкої атлетики, музики та шахів. Серед 250 учнів 56 хлопців та 42 дівчини обрали легку атлетику, 58 хлопців та 37 дівчат — музику, 16 хлопців обрали шахи. В електронній таблиці *Гуртки,* що збережена в папці *Електронні таблиці*, створіть формули для обчислення кількості дівчат, що обрали шахи, та загальної кількості учнів, що обрали кожний гурток.

1. Відкрийте файл *Гуртки*, що збережений у папці *Електронні таблиці*.
2. Для обчислення кількості дівчат, що обрали шахи, слід від загальної кількості всіх учнів відняти суму всіх хлопців, а також дівчат, що обрали інші гуртки. Для створення такої формули виділіть клітинку *C6*, введіть символ «=», клацніть на клітинці *D7*, що містить значення 250, та введіть символ «–». Оберіть інструмент *Вставити функцію* f_x ліворуч від рядка формул і перегляньте список функцій у категорії *Нещодавно використані*. Якщо серед них є функція SUM, оберіть її. В іншому разі оберіть категорію *Математичні* та оберіть цю функцію.
3. Для аргумента *Число 1* виділіть діапазон клітинок *B4*:*B6*, для аргумента *Число 2* — діапазон *C4*:*C5* (мал. 29.5). Натисніть кнопку *OK*.

4. Використовуючи інструмент *Автосума*, створіть формулу в клітинці *D4* для обчислення суми значень із діапазону клітинок *B4*:*C4*.
5. За допомогою автозаповнення скопіюйте створену формулу в клітинки *D5* і *D6*.
6. Збережіть результат у файлі з тим самим іменем у папці *Табличний процесор* власної структури папок.

3. Які математичні та статистичні функції найчастіше використовуються в середовищі табличного процесора?

Найчастіше вживані функції, які можна додати до формули за допомогою засобу *Автосума*, можна знайти також у категоріях *Математичні* або *Статистичні*. Наприклад, функція SUM (СУММ) входить до категорії *Математичні*. Прикладами функцій із категорії *Статистичні* є AVERAGE (СРЗНАЧ) для знаходження середнього значення діапазону клітинок, COUNT (СЧЁТ) — кількості непорожніх клітинок у заданому діапазоні, MAX (МАКС) та MIN (МИН) відповідно — для знаходження найбільшого та найменшого значень.

Для розв'язування задач з алгебри та геометрії можна використовувати вбудовані в табличний процесор математичні функції. Найбільш вживані з них наведено в таблиці 29.1.

Таблиця 29.1

Функція	Функція (рос. інтрефейс)	Результат
ABS(число)	ABS(число)	Модуль (абсолютне значення) аргумента
COS(число)	COS(число)	Косинус аргумента, заданого в радіанах
DEGREES(число)	ГРАДУСЫ(число)	Перетворює значення кута, заданого в радіанах, у градуси
MOD(число; дільник)	ОСТАТ(число; дільник)	Остача від ділення заданого числа на вказаний дільник
SIN(число)	SIN(число)	Синус аргумента, заданого в радіанах
TAN(число)	TAN(число)	Тангенс аргумента, заданого в радіанах
PI()	ПИ()	Значення числа π з точністю до 15 знаків
POWER(число; степінь)	СТЕПЕНЬ(число; степінь)	Результат піднесення числа до вказаного степеня
RADIANS(число)	РАДИАНЫ(число)	Перетворює значення кута, заданого в градусах, у радіани
ROUND(число; кількість знаків)	ОКРУГЛ(число; кількість знаків)	Округлене число до вказаної кількості знаків після коми
SQRT(число)	КОРЕНЬ(число)	Значення арифметичного квадратного кореня аргумента

ДІЄМО

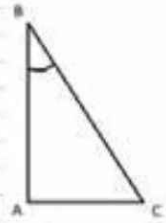
Вправа 4. Використання математичних функцій у формулах.

Завдання. Для прямокутного трикутника *ABC* обчисліть довжину гіпотенузи та другого катета за заданими значеннями катета і протилежного до нього кута.

1. Відкрийте файл *Прямокутний трикутник,* що збережений у папці *Електронні таблиці*.
2. Виділіть клітинку *B5*, на вкладці *Формули* оберіть категорію *Математичні* та функцію RADIANS, для введення аргумента клацніть на клітинці *B4*. Формула в клітинці *B5* набуде вигляду: = RADIANS(*B4*).
3. Для обчислення довжини гіпотенузи слід знайти частку довжини катета й синуса протилежного кута. Для створення формули виділіть клітинку *B8*, введіть символ «=», клацніть на клітинці *B3* та введіть символ «/». Оберіть інструмент *Вставити функцію* f_x, у категорії *Математичні* знайдіть та оберіть функцію SIN. Зверніть увагу, що аргументом цієї функції має бути значення в радіанах, тому в поле *Число* слід ввести посилання на клітинку *B5*. Формула набуде вигляду *=B3*/SIN(*B5*).

   Примітка. Формула може містити кілька функцій, наприклад, якщо не обчислювати значення кута *B* в радіанах у клітинці *B5*, можна записати формулу в клітинці *B8* так: *=B3*/SIN(RADIANS(*B4*)).
4. Аналогічно в клітинці *B9* створіть формулу для обчислення довжини другого катета.
5. Для клітинки *B9* встановіть числовий формат з округленням до десятих (мал. 29.8).
6. Збережіть результат у файлі з тим самим іменем у папці *Електронні таблиці* власної структури папок.

	A	B	C	D
1	У прямокутному трикутнику ABC знайди сторони AB та BC			
2				
3	Катет AC, см	4		
4	Кут B, у градусах	30		
5	Кут B, у радіанах	0,523599		
6				
7				
8	Гіпотенуза BC, см	8		
9	Катет AB, см	6,9		
10				
11				



Мал. 29.8

4. Як у формулах використовують логічні функції?

У середовищі табличного процесора використовуються також формули, що мають аргументом логічний вираз. Логічний вираз містить знак порівняння та може набувати одного з двох значень: TRUE (істина) та FALSE (хибність) залежно від конкретних значень, що зберігаються у клітинках, на адреси яких існує посилання у виразі. Приклади логічних виразів подано у таблиці 29.2.

Таблиця 29.2

Логічний вираз	Пояснення умови, за якої логічний вираз набуває значення TRUE (істина)
A1>1	Числове значення в клітинці *A1* більше за 1
*F2***A4*=SUM(*B2*:*B13*)	Добуток значень у клітинках *F2* та *A4* дорівнює сумі значень діапазону *B2*:*B13*
B13="Петрик"	Вмістом клітинки *B13* є текстове значення «Петрик»
B3–*C3*>=12	Різниця значень у клітинках *B3* та *C3* більше або дорівнює 12

Якщо результат обчислення за формулою залежить від виконання деякої умови — істинності логічного виразу, то використовують **логічні функції**. При складанні алгоритмів у цьому разі ви використовуєте алгоритмічну структуру розгалуження. До категорії логічних функцій належить функція IF (ЕСЛИ), за допомогою якої можна організувати розгалу-

женння в середовищі табличного процесора (мал. 29.9).

Функція IF має три аргументи:
IF(*логічний_вираз*; *значення_якщо_істина*; *значення_якщо_хибність*).

При використанні функції IF перевіряється істинність вказаного логічного виразу, і до тієї клітинки, де міститься формула, заноситься *значення_якщо_істина*, якщо логічний вираз істинний, або *значення_якщо_хибність*, якщо логічний вираз хибний. Аргументами *значення_якщо_істина* та *значення_якщо_хибність* можуть бути число, текстове значення, яке має бути записане в лапках, або вираз для обчислення. Вираз для обчислення може містити інші функції, зокрема і вкладену функцію IF, якщо слід сформувати більш складну умову.

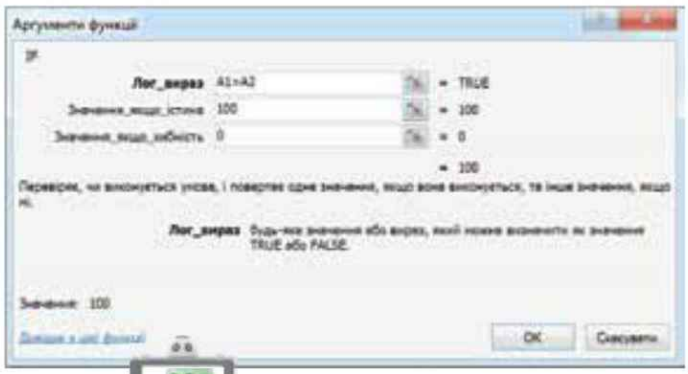
Мал. 29.9

Приклади формул, що містять логічну функцію IF, наведено у таблиці 29.3.

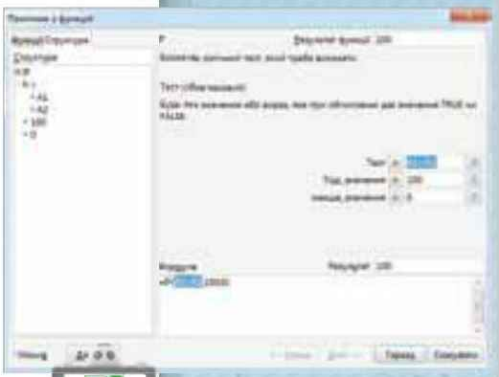
Таблиця 29.3

Формула	Значення в клітинках	Результат обчислення за формулою
=IF(*A1***B2*>0; SQRT(*A1***B2*); POWER(*A1*,*B2*))	*A1* — 2, *B2* — 18	6
	A1 — 5, *B2* — (–1)	0,2
	A1 — (–1), *B2* — 4	1
=IF(*B2*>*C2*;"Перевищення бюджету"; "OK")	*B2* — 450, *C2* — 500	OK
	B2 — 300, *C2* — 250	Перевищення бюджету
=IF(*A10*=100;SUM(*B5*:*B7*);"")	*A10* — 100, *B5* — 32, *B6* — 45, *B7* — 18	95
	A10 — 50, *B5* — 100, *B6* — 210, *B7* — 180	Порожня клітинка
=IF(*A1*>*A2*;100;0)	*A1* — 20, *A2* — 25	0
	A1 — 20, *A2* — 15	100

Під час використання майстра функцій кожний з аргументів функції IF записують в окремому полі (мал. 29.10, *а*, *б*).



Мал. 29.10, *а*



Мал. 29.10, *б*

Для створення складеного логічного виразу використовують інші функції з категорії *Логічні*: AND(*логзнач1*; *логзнач2*;...) (И(*логзнач1*; *логзнач2*;...)), OR(*логзнач1*; *логзнач2*;...) (ИЛИ(*логзнач1*; *логзнач2*;...)), NOT(*логзнач*) (НЕ(*логзнач*)). Аргументами логічних функцій є логічні вирази, що можуть набувати одне з двох значень — істина чи хибність. Їх використання аналогічне відповідним командам, які ви використовуєте для запису складених умов під час складання алгоритмів — програм.

Найчастіше логічні функції AND, OR, NOT використовують для запису складеної умови як аргументу логічної функції IF.

ДІЄМО

Вправа 5. Використання логічної функції IF.

Завдання. Для даних зросту учнів 8 класу, заданих у стовпчику *B* електронної таблиці *Зріст*, у стовпчику *C* створіть формули для визначення кольору маркування шкільних парт за умови, що для учнів, які мають зріст понад 160 см, маркування має бути зеленим, а для інших — червоним.

1. Відкрийте файл *Зріст*, що збережений у папці *Електронні таблиці*.
2. Виділіть клітинку *C2*. Оберіть інструмент *Вставити функцію* f_x, у категорії *Логічні* знайдіть та оберіть функцію IF.
3. Виділіть аргумент *Логічний вираз*, клацніть на клітинці *B2*, введіть із клавіатури <=160.
4. Виділіть аргумент *Значення_якщо_істина*, введіть текст *Червоне*. Зверніть увагу, що текстове значення автоматично буде взято в лапки.
5. Виділіть аргумент *Значення_якщо_хибність*, введіть текст *Зелене*. Натисніть кнопку *OK*. Переконайтеся, що формула має вигляд: =IF(B2<=160;"Червоне";"Зелене")
6. Виділіть клітинку *C2*, скористайтеся автозаповненням для копіювання формули в діапазон клітинок *C3*:*C6*. Перевірте результат у діапазоні *C3*:*C6* (мал. 29.11).
7. Збережіть результати роботи у файлі з тим самим іменем у папці *Табличний процесор* власної структури папок.

	A	B	C
1	Прізвище учня	Зріст	Маркування
2	Антоненко	145	Червоне
3	Байтало	160	Червоне
4	Вівчар	162	Зелене
5	Гнатишин	149	Червоне
6	Едгар	170	Зелене

Мал. 29.11

Мал. 29.12

5. Як скористатися засобом умовного форматування даних?

Змінити в електронній таблиці форматування даних, що відповідають певним умовам, можна за допомогою **умовного форматування**. Цей засіб використовують, щоб створювати правила форматування окремих клітинок таблиці залежно від їх значення.

Наприклад, можна застосувати умовне форматування до клітинок так, щоб будь-яке числове значення, що менше ніж 500, відображалось у відповідних клітинках на червоному фоні. Дія такого умовного форматування відповідає команді розгалуження (мал. 29.12):

Якщо значення в клітинці, що входить до виділеного діапазону, <500,
то застосувати до клітинки форматування: заливка — червоного кольору.

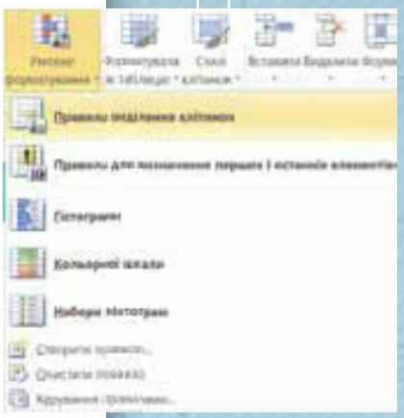
Табличні процесори також дають змогу застосовувати умовне форма-

	A	B	C	D	E	F	G	H
1					*Навчальні досягнення учнів*			
2								
3	*Прізвище*	*Алгебра*	*Геометрія*	*Фізика*	*Інформатика*	*Укр.літ.*	*Заруб.літ.*	*Біологія*
4	Бондар В.	11	11	10	12	10	11	11
5	Гапон С.	9	8	9	8	9	10	10
6	Стецюк К.	8	9	9	9	8	9	10
7	Савицька О.	4	4	3	4	4	3	5
8	Ткаченко В.	9	9	10	11	11	11	10

Мал. 29.13

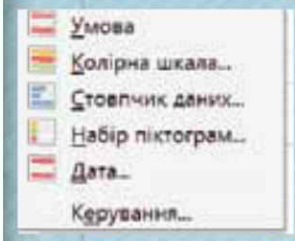
тування з використанням гістограм, кольорових шкал та наборів піктограм. Наприклад, за допомогою кольорової шкали всі найменші значення виділеного діапазону можуть мати червоний колір заливки, найбільші значення — зелений, а всі проміжні значення — інші відтінки від червоного до зеленого (мал. 29.13).

Дія умовного форматування з використанням гістограм, кольорової шкали та набору піктограм відповідає команді розгалуження з вкладеннями. Наприклад, дія умовного форматування з використанням кольорової шкали за шаблоном , результат якого відображено на малюнку 29.13, відповідає команді розгалуження:



Мал. 29.14, *а*

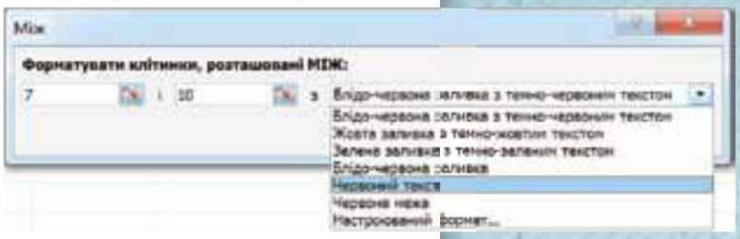
Якщо значення в клітинці збігається з **мінімальним** значенням діапазону,
то застосувати до клітинки форматування: заливка — червоного кольору,
інакше якщо значення в клітинці збігається з **максимальним** значенням діапазону,
то застосувати до клітинки форматування: заливка — зеленого кольору,
інакше застосувати форматування: заливка — відтінки від червоного до зеленого.



Мал. 29.14 *б*

Для умовного форматування вмісту клітинок необхідно виділити клітинки, до яких застосовуватиметься таке форматування: на вкладці *Основне* в групі *Стилі* обрати інструмент *Умовне форматування* (для *Microsoft Excel 2010*) або вибрати в меню *Формат* вказівку *Умовне форматування* (для *LibreOffice Calc*). Після цього слід обрати тип умовного форматування для створення нового правила виділення клітинок або гістограми, кольорові шкали чи набори піктограм (мал. 29.14, *а*,*б*).

Під час створення нового правила виділення клітинок можна обрати умову для форматування, задати значення, з якими слід порівнювати значення з діапазону клітинок, та вказати формат, який необхідно застосувати (мал. 29.15). Якщо серед запропонованих форматів жоден не підходить, можна обрати варіант



Мал. 29.15

3. Клітинка *B2* містить значення — 20, клітинка *C2* — значення 10. У клітинку *D2* введено формулу з використанням логічної функції IF. По черзі визначайте, яке значення буде відображено в клітинці *D2* та обґрунтовуйте свою відповідь.

1) =IF(*B2*<0; –*B2*; *B2*)
2) =IF(*B2*>*C2*; "більше"; "")
3) =IF(*C2*>0; *B2*+*C2*; *B2*–*C2*)

4. У клітинці *A1* введено число 100, а в клітинці *B1* — число 10. По черзі називайте значення логічних функцій та обґрунтовуйте свою відповідь:

1) AND(*A1*<0; *B1*>0)
2) OR(*A1*<0; *B1*>0)
3) AND(*A1*>20; *B1*<=50)
4) NOT(*A1*<>0)
5) AND(*A1*>50; *A1*<>70; *B1*=10)
6) OR(*A1*<200; *B1*<>0; *B1*<20)

ПРАЦЮЄМО САМОСТІЙНО

1. Виконайте обчислення у файлі електронної таблиці *Правило,* що збережений у папці *Електронні таблиці* (мал. 29.17). За допомогою табличного процесора перевірте істинність твердження $a \cdot b = НСК(a; b) \cdot НСД(a; b)$.

Перше число	Друге число	Добуток	Найменше спільне кратне	Найбільший спільний дільник	Добуток НСК та НСД
a	*в*	*ав*	*НСК (а,в)*	*НСД(а,в)*	
48	12				

Мал. 29.17

2. У середовищі табличного процесора складіть модель перевірки умови плавання тіла. Візьміть до уваги: якщо сила тяжіння більша від архімедової сили, тіло буде тонути; якщо сила тяжіння дорівнює силі Архімеда, то тіло може перебувати в рівновазі в будь-якій точці рідини; якщо сила тяжіння менша від архімедової сили, тіло буде спливати, підніматися вгору. Як обчислювати силу Архімеда та силу тяжіння, пригадайте з уроків фізики або знайдіть в Інтернеті.

3. У файлі *Каса,* що збережений у папці *Електронні таблиці,* розміщено зведені відомості про продаж квитків касами автовокзалу. Знайдіть вартість квитків по кожному напрямку та загалом. Застосуйте умовне форматування *блідо-червона заливка та темно-червоний текст* для позначення перших п'яти значень із найбільшою кількістю проданих квитків та у вигляді гістограми для суми від продажу квитків (мал. 29.18).

Відправлення	Маршрут	Прибуття	Ціна	Кількість проданих квитків	Сума
5:00	ТЕРНОПІЛЬ АВ	7:20	37,91	27	1023,57
04.01.2016	- ЖАДОВА				
5:10	ТЕРНОПІЛЬ АВ	7:40	50,81	19	965,39
04.01.2016	- МОНАСТИРИСЬКА				
5:15	ТЕРНОПІЛЬ АВ	8:45	93,66	25	2341,5
04.01.2016	- РІВНЕ				
5:15	ТЕРНОПІЛЬ АВ	8:05	88,93	27	2401,11
04.01.2016	- ІВ.ФРАНКІВСЬК 2				
5:20	КИЇВ АВ	10:30	100,45	25	2511,25
04.01.2016	- ЯРЕМЧА				

Мал. 29.18

4. Відкрийте файл *Квадратний корінь*, збережений у папці *Електронні таблиці.* Створіть формулу для обчислення значень арифметичного квадратного кореня для невід'ємних чисел. Для від'ємних значень передбачте текстовий коментар «*Не існує*». Скористайтеся логічною функцією IF та функцією SQRT з категорії *Математичні*.

5. Створіть електронну таблицю для обчислення вартості поїздки автомобілем *Nissan Note* від Києва до вказаних на малюнку 29.19 міст. Відстань можна дізнатися з Інтернету чи розрахувати за картою автомобільних доріг України, враховуючи відповідний її масштаб. Вартість 1 л пального та витрати палива на 100 км для даного автомобіля знайдіть в Інтернеті.

	A	B	C
1	Дорога до відпочинку		
2			
3	Родина збирається у відпустку. Підрахуйте, скільки необхідно витратити коштів на білети за умови, що:		
4			
5	1. Їдуть 2 дорослих і 1 дитина.		
6	2. Вартість дорослого квитка – 130 у.о.		
7	3. Дітям — знижка в 40 %.		
8	4. 1 у.о. = 23,54 грн.		
9		Повний квиток	130
10		Дитячий квиток	78
11		Разом в у.о.	338
12		Разом в грн.	7956,52

Мал. 29.19

Мал. 29.20

ДОСЛІДЖУЄМО

1. Дослідіть, які набори піктограм можна використовувати для умовного форматування даних у таблиці. Для цього введіть у таблиці значення 10, 20, 30, 40, виділіть діапазон клітинок із цими даними, оберіть інструмент *Умовне форматування* на вкладці *Основне* та *Набори піктограм* (для *Microsoft Excel 2010*) або вказівку *Формат/Умовне форматування/Набір піктограм* (для *LibreOffice Calc*). Обирайте різні набори піктограм та визначте, як змінюється відображення даних для кожного з наборів (мал. 29.20).

2. Дослідіть, яку кількість умов можна використовувати одночасно при застосуванні умовного форматування.

3. Дослідіть, правила яких типів можна створювати для умовного форматування даних таблиці.

30. ПРАКТИЧНА РОБОТА 14

РОЗВ'ЯЗУВАННЯ ЗАДАЧ НА ОБЧИСЛЕННЯ

ПРИГАДАЙТЕ

- як записувати абсолютні, відносні та комбіновані посилання у формулах;
- як використовувати автозаповнення для копіювання формул;
- як застосовувати вбудовані функції для опрацювання даних.

СТВОРІТЬ

У власній структурі папок створіть папку *Практична робота 14.*

ПАМ'ЯТАЙТЕ

Під час виконання практичних завдань пам'ятайте про правила безпеки життєдіяльності при роботі з комп'ютером!

Завдання 1. Степені натуральних чисел (*8 балів*)

Засобами табличного процесора створіть електронну таблицю степенів натуральних чисел першого десятка від першого степеня до п'ятого.

Завдання 2. Об'єм газу (*8 балів*)

Засобами табличного процесора створіть електронну таблицю для визначення об'єму газу за нормальних умов.

	A	B	C	D	E	F
1	Молярний об'єм 1 моля газу за нормальних умов, л/моль					22,4
2	Газ	Маса газу, г	Молярна маса, г/моль	Об'єм, л		
3	Кисень	0,6	32		=B3/C3*F1	
4	Водень	0,4	2			
5	Вуглекислий газ	1,8	44			
6						

Завдання 3. Енергоспоживання (*10 балів*)

Засобами табличного процесора створіть електронну таблицю для планування економії споживання електричної енергії. Скористайтеся файлом *Енергоспоживання* для отримання відомостей про потужність побутового пристрою. Обчисліть обсяг споживання електроенергії на тиждень у вашій родині та її вартість. Сплануйте, де в підрахунках мають бути абсолютні, а де — відносні посилання.

Завдання 4. Поїздка (*11 балів*)

Створіть засобами табличного процесора електронну таблицю, структуру і формат якої зображено на малюнку, та збережіть її з іменем *Поїздка* в папці *Практична робота 14*. У клітинки *C9*:*C12* введіть дані та формули відповідно до завдання, яке записане у клітинці *A4* таблиці.

	A	B	C
1	Дорога до відпочинку		
2			
3	Родина збирається у відпустку		
4	Підрахуйте, скільки необхідно витратити коштів на білети за умови, що:		
5	1. Їдуть 2 дорослих і 1 дитина.		
6	2. Вартість дорослого квитка - 130 у.о.		
7	3. Дітям - знижка у 40%.		
8	4. 1 у.о. = 23,54 грн.		
9		Повний квиток	130
10		Дитячий квиток	78
11		*Всього в у.о.*	338
12		*Всього в грн.*	7956,52
13			

Завдання 5. Закон Кулона (*11 балів*)

Засобами табличного процесора створіть структуру розрахункової таблиці для лабораторної роботи з фізики для визначення електростатичної сили взаємодії двох заряджених частинок за законом Кулона за зразком та заповніть її. Візьміть до уваги, що закон Кулона має вигляд: $F_{12} = k \cdot \frac{q_1 \cdot q_2}{r_{12}^2}$. Передбачте, які посилання на клітинки можна використати для розрахунків.

A	B	C	D	E
		Закон Кулона		
Заряд1	Заряд 2	Відстань між зарядами	Електростатична сила взаємодії	Електростатична стала
q1(Кл)	q2(Кл)	r(м)	F	k(10^9 Н·м2·Кл-2)
0,0002	0,0000016	0,01	28760,7758	8,987742438
0,000008	0,0007	0,5	201,3254306	
0,00074	0,00008	0,001	532074352,3	