**Тема уроку: Відображення базових графічних примітивів – лінія, прямокутник, сектор, ламана, еліпс, налаштування стилю та кольору примітивів засобами мови програмування.**

**Актуалізація опорних знань учнів**

* Які графічні примітиви можна створювати у графічному редакторі?
* Які графічні примітиви можна створювати у текстовому процесорі?
* Назвіть властивості графічних примітивів.

**Мотивація навчальної діяльності**

Сприйняття візуальної інформації є основним для людини. Не секрет, що 90% інформації людина сприймає саме через зір. Вміння створювати графічні зображення є важливою компетенцією сучасної людини.

Діти, Ви же вмієте створювати графічні примітиви в різних прикладних програмах. Прийшов час навчитися створювати їх засобами мови програмування C# в середовищі Visual Studio 2015.

**V. Вивчення нового матеріалу.**

Перш ніж що-небудь намалювати у вікні форми треба створити об’єкт класу Graphics та провести його ініціалізацію. Наприклад таким чином:

Graphics g;

g = CreateGraphics();

Клас Pen визначає об'єкт, який використовується для малювання прямих ліній і кривих.

Приклади :

Pen myPen1 = new Pen(Color.Gold, 2); // перший аргумент – колір, другий – товщина.

Pen myPen1 = new Pen(Color.Gold);// тільки один аргумент, що визначає колір.

Клас SolidBrush визначає пензлик одного кольору. Пензлики використовуються для заливки графічних фігур, таких як прямокутники, еліпси, кола, багатокутники.

Приклад:

SolidBrush b1;

b1 = new SolidBrush(Color.LightGreen)

Перш ніж будувати графічні примітиви подумайте, де знаходиться початок координат та куди напрямлені осі.

Відповідь. Додаток Windows Forms визначає положення вікна на екрані в екранних координатах. Початок екранних координат знаходиться у верхньому лівому куті екрана, осі напрямлені вниз.

**Методи для побудови базових графічних примітивів.**

Метод DrawLine проводить відрізок, що сполучає дві точки, які задаються парами координат( (x1, y1) - координати одного кінця; (x2, y2) - координати другого кінця). Загальний вигляд DrawLine (Pen, x1, y1, x2, y2).

Приклад:

Pen myPen1 = new Pen(Color.Gold, 2);

gr.DrawLine (myPen1, 50, 100, 200, 300)

Метод DrawRectangle малює прямокутник, який визначається парою координат (x1,y1) (координати лівого верхнього кута прямокутника), ширини w і висоти h. Якщо ширина і висота співпадають, то отримаємо квадрат. Загальний вигляд:  [DrawRectangle (Pen, x1, y1, w, h)](https://msdn.microsoft.com/ru-ru/library/wt06kxfd(v=vs.110).aspx), Перший аргумент об’єкт класу Pen.

Приклад:

Pen p1 = new Pen(Color.Blue, 3);

DrawRectangle (Pen, 100, 200, 70, 70)

Метод FillRectangle заповнює внутрішню частину прямокутника, що визначається обмежуючим прямокутником, заданим за допомогою пари координат (x1, y1) (координати лівого верхнього кута прямокутника), ширини w і висоти h. Загальний вигляд: [FillRectangle (Brush, x1, y1, w, h)](https://msdn.microsoft.com/ru-ru/library/wt06kxfd(v=vs.110).aspx), Перший аргумент – об’єкт класу SolidBrush.

Приклад:

SolidBrush b1;

b1 = new SolidBrush(Color.Green);

graph.FillRectangle(b1, 170, 130, 70, 140)

Метод DrawEllipse малює еліпс, який визначається обмежуючим прямокутником, заданим за допомогою пари координат (x1,y1) (координати лівого верхнього кута прямокутника), ширини w і висоти h. Якщо ширина і висота співпадають, то отримаємо круг. Загальний вигляд: [DrawEllipse(Pen, x1, y1, w, h)](https://msdn.microsoft.com/ru-ru/library/wt06kxfd(v=vs.110).aspx), Перший аргумент об’єкт класу Pen.

Приклад:

Pen myPen1 = new Pen(Color.Gold, 2);

gr.DrawEllipse(myPen1, 200, 200, 300, 150)

Метод FillEllipse заповнює внутрішню частину еліпса, що визначається обмежуючим прямокутником, заданим за допомогою пари координат (x1, y1) (координати лівого верхнього кута прямокутника), ширини w і висоти h. Загальний вигляд:  [FillEllipse (Brush, x1, y1, w, h)](https://msdn.microsoft.com/ru-ru/library/wt06kxfd(v=vs.110).aspx), Перший аргумент – об’єкт класу SolidBrush.

Приклад:

SolidBrush b1;

b1 = new SolidBrush(Color.LightGreen);

gr.FillEllipse(b1, 200, 200, 300, 150)

Метод DrawArc малює дугу, яка є частиною еліпса, заданого парою координат, шириною і висотою. Загальний вигляд: DrawArc (Pen, x, y, w, h, a, l), a – початкове значення кута (в градусах), який вимірюється за годинниковою стрілкою, починаючи від осі X; l – розмір дуги.

Pen myPen1 = new Pen(Color.Gold, 2);

gr.DrawArc(myPen1, 445, 380, 140, 140, 180, 180)

Клас FillPie заповнює внутрішню частину сектора, що визначається еліпсом, який заданий парою координат, шириною, висотою і двома кутами. Загальний вигляд: FillPie (Pen, x, y, w, h, a, l), a – початкове значення кута (в градусах), який вимірюється за годинниковою стрілкою, починаючи від осі X; l – розмір дуги.

Приклад:

b1 = new SolidBrush(Color.Gold);

gr.FillPie(b1, 445, 380, 140, 140, 180, 180)

Для побудови багатокутника треба задати масив точок з координатами, що обмежують багатокутник, а потім побудувати контур багатокутника та (або) заливку.

Для побудови контура багатокутника використовується метод DrawPolygon. Загальний вигляд: DrawPolygon(myPen1, myPoints), myPoints – масив точок.

Для заливки внутрішньої області багатокутника використовується метод

FillPolygon. Загальний вигляд: FillPolygon(b1, myPoints), myPoints – масив точок.

Приклад:

Pen myPen1 = new Pen(Color.Gold, 2);

b1 = new SolidBrush(Color.Gold);

// масив точок

Point[] myPoints4 =

{

new Point(550, 210),

new Point(550, 340),

new Point(500, 275),

};

//багатокутник

gr.FillPolygon(b1, myPoints4);

gr.DrawPolygon(myPen1, myPoints4);

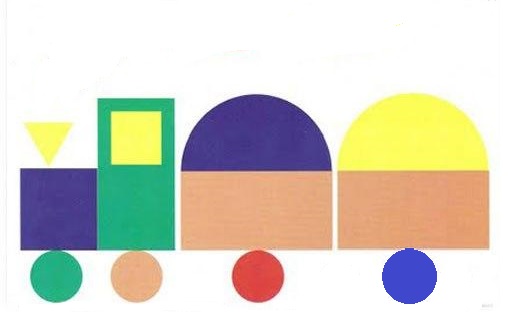
Давайте повторимо правила поведінки та безпеки в комп’ютерному класі.

Увага! Під час роботи за комп’ютером дотримуйтеся правил безпеки і санітарно-гігієнічних норм.

Зараз перейдемо до виконання практичної роботи.

**VI. Практична робота.**

Завдання1. Скласти програму, яка будує зображення машини за поданим зразком:



Хід виконання:

1. Створити на робочому столі власну папку.
2. Запустити середовище програмування Visual Studio 2015.
3. Створити проект WindowsForms та зберегти у власну папку.
4. Розмістити на формі один елемент Button. Задати властивості форми та кнопки самостійно.
5. Створити подію Click для кнопки.
6. Задати код для кнопки «Намалювати»

private void button1\_Click(object sender, EventArgs e)

{

Graphics graph;

SolidBrush b1;

graph = CreateGraphics();

b1 = new SolidBrush(Color.Blue);

graph.FillRectangle(b1, 100, 200, 70, 70);

b1 = new SolidBrush(Color.Green);

graph.FillRectangle(b1, 170, 130, 70, 140);

b1 = new SolidBrush(Color.Yellow);

graph.FillRectangle(b1, 180, 140, 50, 50);

b1 = new SolidBrush(Color.Green);

graph.FillEllipse(b1, 115, 270, 40, 40);//перше колесо

b1 = new SolidBrush(Color.Gold);

graph.FillEllipse(b1, 185, 270, 40, 40); //друге колесо

b1 = new SolidBrush(Color.LightCoral);

graph.FillRectangle(b1, 245, 200, 120, 70); //перший причіп

b1 = new SolidBrush(Color.LightCoral);

graph.FillRectangle(b1, 370, 200, 120, 70); //другий причіп

b1 = new SolidBrush(Color.LightPink);

graph.FillEllipse(b1, 285, 270, 40, 40); //третє колесо

b1 = new SolidBrush(Color.Gold);

graph.FillEllipse(b1, 410, 270, 40, 40); //четверте колесо

b1 = new SolidBrush(Color.Gold);

graph.FillPie(b1, 245, 165, 120, 70, 180, 180); //перший вантаж

b1 = new SolidBrush(Color.Gold);

graph.FillPie(b1, 370, 165, 120, 70, 180, 180); //другий вантаж

b1 = new SolidBrush(Color.Gold);

Point[] myPoints =

{

new Point(110, 185),

new Point(130, 185),

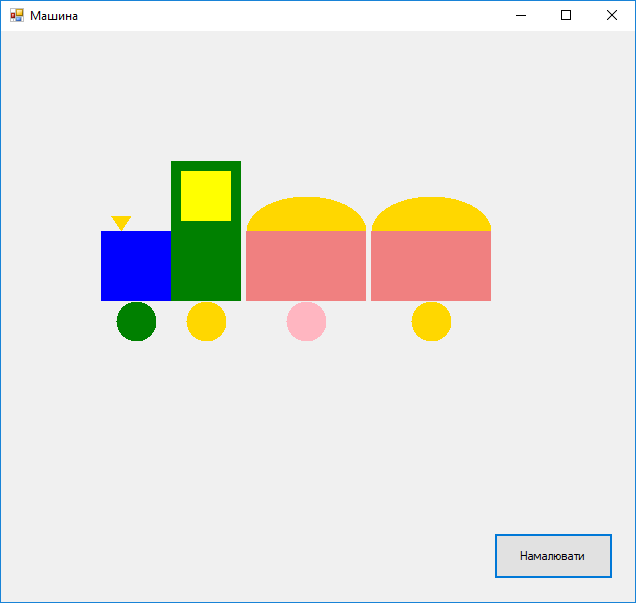
new Point(120, 200),

};

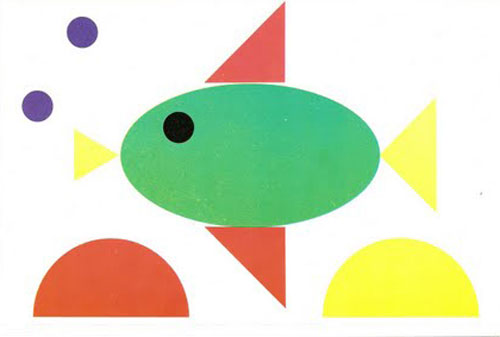
graph.FillPolygon(b1, myPoints); //побудова та заливка трикутника

}

1. Запустити проект на виконання. Очікуваний результат:



Завдання2. Скласти програму, яка будує зображення рибки за поданим зразком:



Хід виконання:

1. Створити проект WindowsForms та зберегти у власну папку.
2. Розмістити на формі один елемент Button. Задати властивості форми та кнопки самостійно.
3. Створити подію Click для кнопки.
4. Задати код для кнопки «Малюй!»

private void button1\_Click(object sender, EventArgs e)

{

Graphics gr;

gr = CreateGraphics();

Pen myPen1 = new Pen(Color.Gold, 2);

SolidBrush b1;

b1 = new SolidBrush(Color.LightGreen);

gr.FillEllipse(b1, 200, 200, 300, 150);//тулуб риби

gr.DrawEllipse(myPen1, 200, 200, 300, 150);

b1 = new SolidBrush(Color.Red);

Point[] myPoints1 =

{

new Point(310, 200),

new Point(390, 200),

new Point(390, 120),

};

//graph.DrawPolygon(myPen1, myPoints);

gr.FillPolygon(b1, myPoints1); //заливка трикутника

// масив точок з координатами нижнього плавника

Point[] myPoints2 =

{

new Point(310, 350),

new Point(390, 350),

new Point(390, 430),

};

//нижній плавник

gr.FillPolygon(b1, myPoints2);

gr.DrawPolygon(myPen1, myPoints1);

gr.DrawPolygon(myPen1, myPoints2);

//око

b1 = new SolidBrush(Color.Blue);

gr.FillEllipse(b1, 250, 250, 30, 30);

gr.DrawEllipse(myPen1, 250, 250, 30, 30);

myPen1 = new Pen(Color.GreenYellow, 2);

b1 = new SolidBrush(Color.Gold);

// масив точок з координатами носа

Point[] myPoints3 =

{

new Point(170, 300),

new Point(170, 250),

new Point(200, 275),

};

//ніс

gr.FillPolygon(b1, myPoints3);

gr.DrawPolygon(myPen1, myPoints3);

b1 = new SolidBrush(Color.Gold);

// масив точок з координатами хвоста

Point[] myPoints4 =

{

new Point(550, 210),

new Point(550, 340),

new Point(500, 275),

};

//хвіст

gr.FillPolygon(b1, myPoints4);

gr.DrawPolygon(myPen1, myPoints4);

//бульбашки

b1 = new SolidBrush(Color.Violet);

gr.FillEllipse(b1, 100, 210, 30, 30);

gr.FillEllipse(b1, 150, 140, 30, 30);

//морський пейзаж

b1 = new SolidBrush(Color.Gold);

gr.FillPie(b1, 445, 380, 140, 140, 180, 180);

gr.DrawArc(myPen1, 445, 380, 140, 140, 180, 180);

b1 = new SolidBrush(Color.Red);

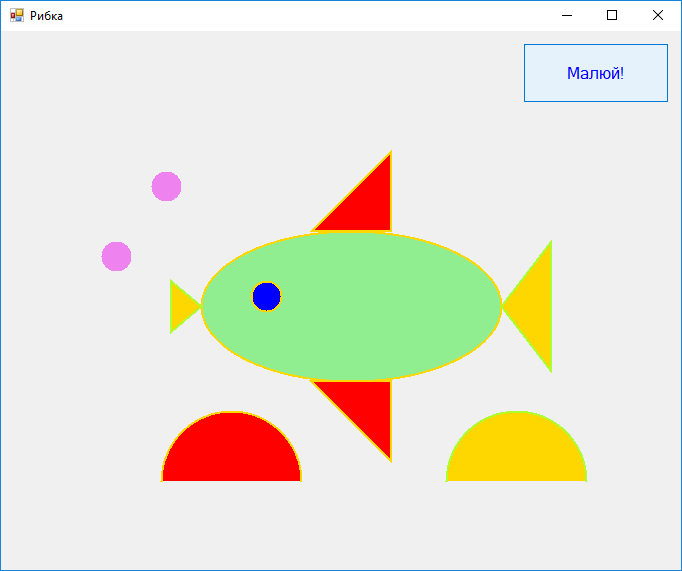
myPen1 = new Pen(Color.Gold, 2);

gr.FillPie(b1, 160, 380, 140, 140, 180, 180);

gr.DrawArc(myPen1, 160, 380, 140, 140, 180, 180);

}

1. Запустити проект на виконання. Очікуваний результат:



**IX. Домашнє завдання. Створити проект, який будує наступне зображення.**

