

BÀI THỰC HÀNH SỐ 5 (4 tiết)

ĐỒ HỌA VỚI GDI+

I. Mục tiêu:

- Giúp sinh viên làm quen với công cụ đồ họa GDI+ trong .Net framework..
- Sử dụng các lớp trong namespace System.Drawing để vẽ và tô màu các hình cơ bản.
- Biết cách xử lý sự kiện chuột: Click, MouseDown, MouseUp, MouseMove...
- Sử dụng các điều khiển Windows Form

II. Hướng dẫn thực hành:

Đối tượng Graphics: Công cụ vẽ

Để tạo đối tượng Graphics, bạn dùng một trong những cách sau đây:

- Lấy Graphics từ một Windows Control qua phương thức CreateGraphics
`public Graphics CreateGraphics();`
- Lấy Graphics từ một Windows Control thông qua trình điều khiển Control
`public static Graphics FromHwnd(IntPtr hwnd);`
Ví dụ: `Graphics graph = Graphics.FromHwnd(this.Handle);`
- Nếu đang vẽ trong hàm xử lý sự kiện Paint, sử dụng thuộc tính Graphics của đối số PaintEventArgs.

```
private void Form1_Paint(object sender, PaintEventArgs e)
{
    Graphics graph = e.Graphics;
    e.Graphics. ...
}
```

Màu sắc (Colors)

Để tạo màu vẽ hoặc màu tô, bạn sử dụng cấu trúc Color với các phương thức sau:

```
public static Color FromArgb(int argb);
public static Color FromArgb(int alpha, Color baseColor);
public static Color FromArgb(int red, int green, int blue);
public static Color FromArgb(int alpha, int red, int green, int blue);
```

Giá trị alpha là độ trong suốt của màu. Red, Green, Blue là các thành phần cơ bản để tạo nên một màu bất kỳ. Các đối số này có giá trị từ 0 đến 255.

Nếu biết rõ tên màu muốn lấy, có thể sử dụng một trong hai phương thức sau:

```
public static Color FromName(string colorName);
public static Color FromKnownColor(KnownColor colorName);
```

Ngoài ra, để đơn giản hơn, cấu trúc Color cung cấp sẵn các màu thông dụng qua các thuộc tính static của nó.

Ví dụ:

`panel1.BackColor = Color.Yellow;`

Bút vẽ (Pens)

Công cụ vẽ cơ bản nhất là bút vẽ. Thư viện GDI+ cung cấp một lớp gọi là Pen. Để tạo một bút vẽ, bạn có thể khai báo một biến kiểu Pen sử dụng một trong các phương thức tạo lập của lớp này.

```
public Pen(Color color);
public Pen(Color color, float width);
```

Hoặc nếu chỉ quan tâm tới màu vẽ, bạn có thể sử dụng các bút vẽ xây dựng sẵn trong lớp Pens.

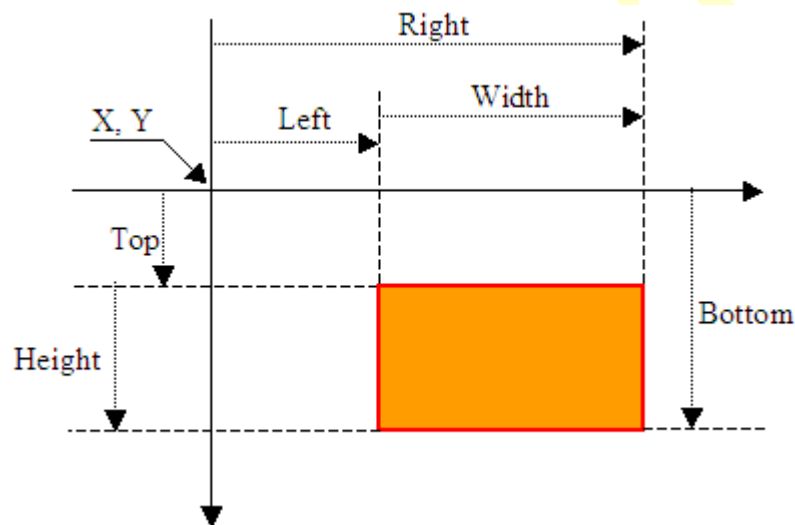
```
Pen penBorder = Pens.Lavender;
```

Vẽ hình chữ nhật và hình vuông

Muốn vẽ một hình chữ nhật kín, bạn dùng phương thức Graphics.DrawRectangle() theo cú pháp sau:

```
public void DrawRectangle(Pen pen, Rectangle rect);
public void DrawRectangle(Pen pen, RectangleF rect);
public void DrawRectangle(Pen pen, int x, int y, int width, int height);
public void DrawRectangle(Pen pen, float x, float y, float wd, float hg);
```

Lưu ý, một hình chữ nhật trong đồ họa máy tính được xác định như sau



Muốn vẽ nhiều hình chữ nhật, bạn sử dụng phương thức Graphics.DrawRectangles() theo một trong hai cách sau:

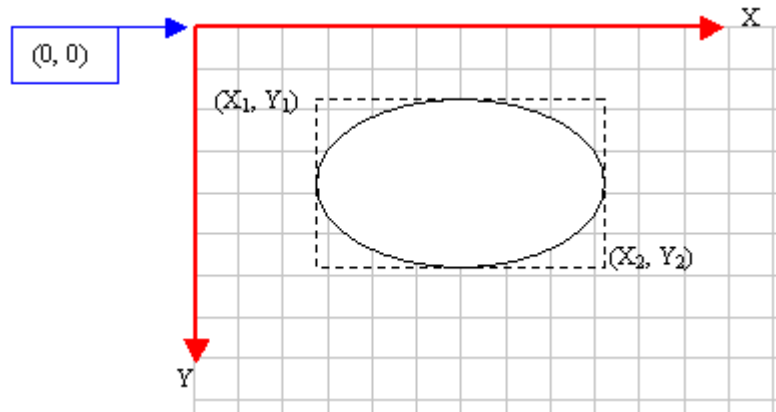
```
public void DrawRectangles(Pen pen, Rectangle[] rects);
public void DrawRectangles(Pen pen, RectangleF[] rects);
```

Ellipse và đường tròn

Để vẽ hình Elip hoặc hình tròn, bạn có thể dùng phương thức Graphics.DrawEllipse() theo một trong 4 dạng sau:

```
public void DrawEllipse(Pen pen, Rectangle rect);
public void DrawEllipse(Pen pen, RectangleF rect);
public void DrawEllipse(Pen pen, int x, int y, int width, int height);
public void DrawEllipse(Pen pen, float x, float y, float width, float height);
```

Các đối số của phương thức này có ý nghĩa như trong phương thức Graphics.DrawRectangle().



Vẽ đoạn thẳng

Để vẽ đoạn thẳng, lớp Graphics được trang bị phương thức DrawLine() với 4 dạng sau:

```
public void DrawLine(Pen pen, Point pt1, Point pt2);
public void DrawLine(Pen pen, PointF pt1, PointF pt2);
public void DrawLine(Pen pen, int x1, int y1, int x2, int y2);
public void DrawLine(Pen pen, float x1, float y1, float x2, float y2);
```

Đường gấp khúc

Để vẽ nhiều đoạn thẳng một lúc, bạn có thể sử dụng phương thức Graphics.DrawLines().

Phương thức này có 2 dạng quá tải sau:

```
Public void DrawLines(Pen pen, Point[] points);
Public void DrawLines(Pen pen, PointF[] points);
```

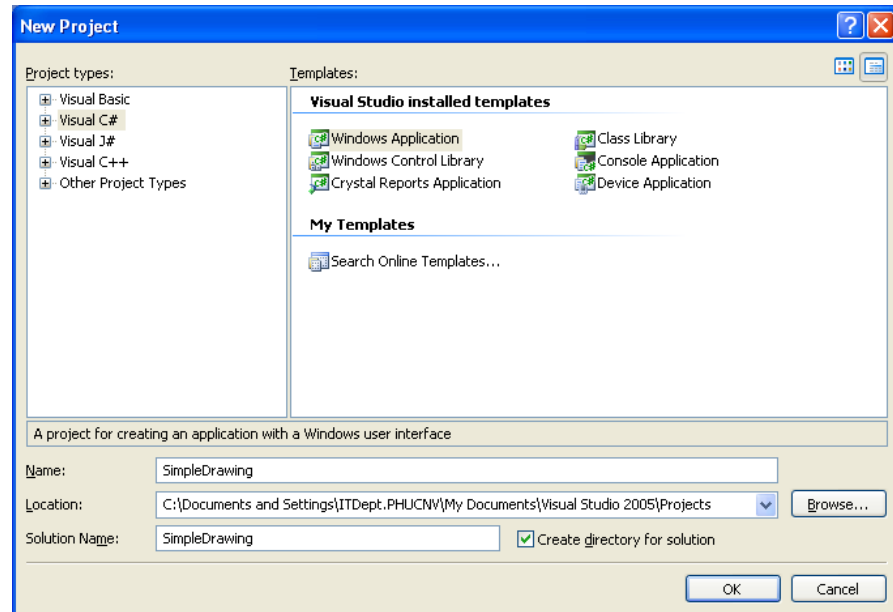
Vẽ đa giác

Để vẽ một đa giác, bạn dùng phương thức Graphics.DrawPolygon() với 2 dạng sau:

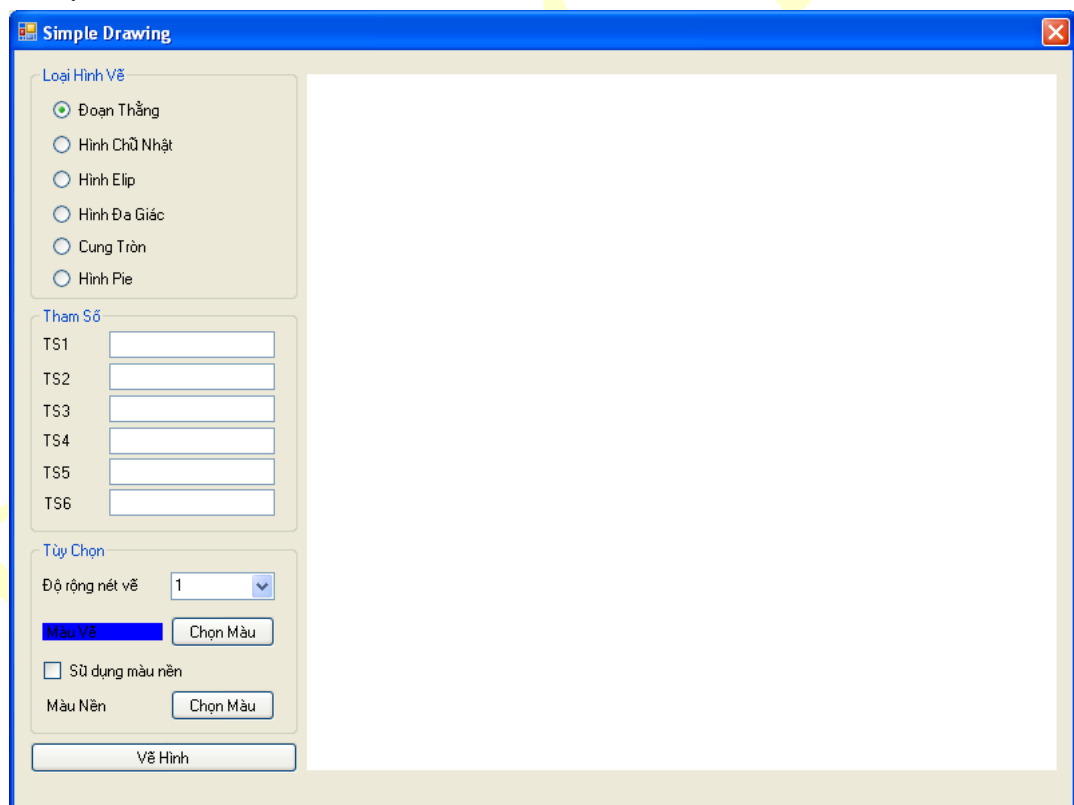
```
Public void DrawPolygon(Pen pen, Point[] points);
Public void DrawPolygon(Pen pen, PointF[] points);
```

III. Bài tập:

1. Tạo dự án Windows Application và đặt tên là SimpleDrawing như sau:

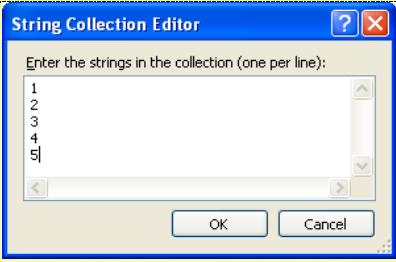


2. Thiết kế lại Form1 theo mẫu sau:



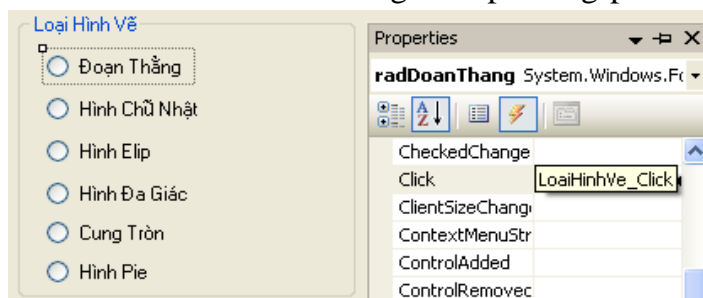
STT	Loại Control	Tên Thuộc Tính	Giá Trị
1	Form1	Text	Simple Drawing
		MaximizeBox	False
		MinimizeBox	False

		MaximumSize	800,600
		MinimumSize	800,600
		Size	800,600
2	GroupBox	Name	grbLoaiHinh
		Text	Loại Hình Vẽ
		Location	12,12
		Size	200,174
3	GroupBox	Name	grbThamSo
		Text	Tham Số
		Location	12,192
		Size	200,268
4	GroupBox	Name	grbTuyChon
		Text	Tùy Chọn Màu
		Location	12,367
		Size	200,144
5	RadioButton	Name	radDoanThang
		Text	Đoạn Thẳng
		Nằm trong Control	grbLoaiHinh
6	RadioButton	Name	radChuNhat
		Text	Hình Chữ Nhật
		Nằm trong Control	grbLoaiHinh
7	RadioButton	Name	radEllipse
		Text	Hình Elip
		Nằm trong Control	grbLoaiHinh
8	RadioButton	Name	radDaGiac
		Text	Hình Đa Giác
		Nằm trong Control	grbLoaiHinh
9	RadioButton	Name	radCungTron
		Text	Cung Tròn
		Nằm trong Control	grbLoaiHinh
10	RadioButton	Name	radPie
		Text	Hình Pie
		Nằm trong Control	grbLoaiHinh
Các Controls sau đây nằm trong GroupBox: grbThamSo			
11	Label	Name	lblTS1
		Text	TS1
12	Label	Name	lblTS2
		Text	TS2
13	Label	Name	lblTS3

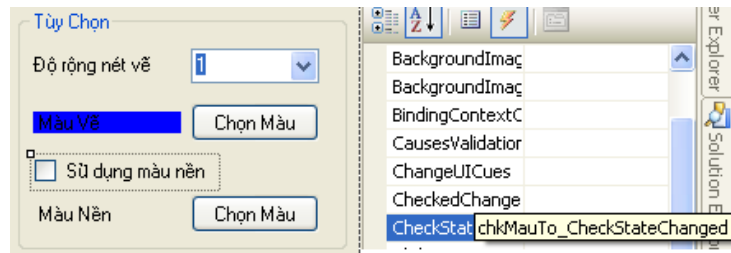
		Text	TS3
14	Label	Name	lblTS4
		Text	TS4
15	Label	Name	lblTS5
		Text	TS5
16	Label	Name	lblTS6
		Text	TS6
17	TextBox	Name	txtThamSo1
		Size	124,20
		Location	59,18
18	TextBox	Name	txtThamSo2
		Size	124,20
		Location	59,42
19	TextBox	Name	txtThamSo3
		Size	124,20
		Location	59,66
20	TextBox	Name	txtThamSo4
		Size	124,20
		Location	59,90
21	TextBox	Name	txtThamSo5
		Size	124,20
		Location	59,113
22	TextBox	Name	txtThamSo6
		Size	124,20
		Location	59,136
Những Controls sau đây nằm trong GroupBox: grpTuyChon			
23	Label	Name	lblKTButVe
		Text	Độ rộng nét vẽ
24	ComboBox	Name	cbbKTButVe
		Text	1
		Items	
25	Label	Name	lblMauVe
		Text	Blue
		AutoSize	False

		Size	90,13
		BackColor	Blue
26	Label	Name	lblMauNen
		Text	Màu Tô
		AutoSize	False
		Size	90,13
		Enabled	False
27	Button	Name	btnMauVe
		Text	Chọn Màu
28	Button	Name	btnMauNen
		Text	Chọn Màu
		Enabled	False
29	CheckBox	Name	chkMauTo
		Text	Tô màu nền cho hình
Nút Vẽ Hình			
30	Button	Name	btnVeHinh
		Text	Vẽ Hình
		Location	12,485
		Size	200,23
Khung Chứa Hình Vẽ			
31	Panel	Name	pnlHinhVe
		Size	561,520
		Location	219,18
		BackColor	White
Các hộp thoại			
32	ColorDialog	Name	colorDialog1

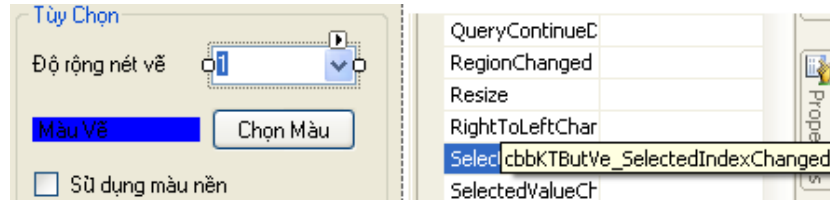
3. Thêm sự kiện Click cho các RadioButton trong GroupBox: grpLoaiHinh



4. Thêm sự kiện CheckStateChanged cho CheckBox: chkMauTo



5. Thêm sự kiện SelectedIndexChanged cho ComboBox: cbbKTButVe



6. Thêm sự kiện Click cho các Button: btnMauVe, btnMauTo, btnVeHinh

7. Thêm sự kiện Load cho Form1.

```
private void LoaiHinhVe_Click(object sender, EventArgs e)
{
}

private void chkMauTo_CheckedChanged(object sender, EventArgs e)
{
}

private void btnMauVe_Click(object sender, EventArgs e)
{
}

private void btnMauNen_Click(object sender, EventArgs e)
{
}

private void btnVeHinh_Click(object sender, EventArgs e)
{
}

private void cbbKTButVe_SelectedIndexChanged(object sender, EventArgs e)
{
    |
}
```

8. Trong lớp Form1.cs, tạo enum LoaiHinh và khai báo thêm các biến như sau:


```

public partial class Form1 : Form
{
    enum LoaiHinh
    {
        DoanThang,
        HinhChuNhat,
        Ellipse,
        DaGiac,
        CungTron,
        Pie
    }

    // =====
    // Khai báo các biến
    // =====

    private Pen    butVe;           // Bút vẽ
    private Brush  coToMau;         // Cọ tô màu
    private Graphics grp;           // Công cụ vẽ
    private LoaiHinh kieuHinh;      // Ghi nhớ kiểu hình muốn vẽ

    // =====
    // Phương thức khởi tạo của lớp Form1
    // =====

    public Form1()
    {
        InitializeComponent();
    }
}

```

9. Trong hàm xử lý sự kiện Form1_Load, thêm đoạn mã sau:

```

private void Form1_Load(object sender, EventArgs e)
{
    // Khởi tạo bút vẽ có màu xanh, độ rộng nét là 1.
    butVe = new Pen(Color.Blue, 1.0f);

    // Khởi tạo màu nền là không màu
    coToMau = new SolidBrush(Color.Transparent);

    // Lấy công cụ vẽ từ Panel: pnlHinhVe
    grp = pnlHinhVe.CreateGraphics();

    // Đánh dấu kiểu hình muốn vẽ mặc định là Đoạn Thẳng
    kieuHinh = LoaiHinh.DoanThang;
}

```

10. Trong hàm xử lý sự kiện btnMauVe_Click, thêm đoạn mã sau:

```
private void btnMauVe_Click(object sender, EventArgs e)
{
    // Hiển thị hộp thoại chọn màu
    if (colorDialog1.ShowDialog() == DialogResult.OK)
    {
        // Đặt lại màu cho bút vẽ
        butVe.Color = colorDialog1.Color;

        // Hiển thị màu lên label Màu vẽ
        lblMauVe.BackColor = colorDialog1.Color;

        // Hiển thị tên màu đã chọn
        lblMauVe.Text = colorDialog1.Color.Name;
    }
}
```

11. Trong hàm xử lý sự kiện btnMauNen_Click, thêm đoạn mã sau:

```
private void btnMauNen_Click(object sender, EventArgs e)
{
    // Hiển thị hộp thoại chọn màu
    if (colorDialog1.ShowDialog() == DialogResult.OK)
    {
        // Thay đổi màu cho cọ vẽ
        coToMau = new SolidBrush(colorDialog1.Color);

        // Hiển thị màu lên label Màu nền
        lblMauNen.BackColor = colorDialog1.Color;

        // Hiển thị tên màu đã chọn
        lblMauNen.Text = colorDialog1.Color.Name;
    }
}
```

12. Trong hàm xử lý sự kiện chkMauTo_CheckedChanged, thêm đoạn mã sau:

```
private void chkMauTo_CheckedChanged(object sender, EventArgs e)
{
    // Nếu checkbox "Sử dụng màu nền" được chọn
    // thì hiển thị label và nút chọn màu nền.
    // ngược lại thì làm mờ để cấm sử dụng

    lblMauNen.Enabled = chkMauTo.Checked;
    btnMauNen.Enabled = chkMauTo.Checked;

    // Nếu checkbox được chọn thì phải hiển thị màu tô
    if (chkMauTo.Checked)
    {
        // Lấy màu tô từ cọ vẽ và đặt làm màu nền cho label
        lblMauNen.BackColor = ((SolidBrush) coToMau).Color;
    }
}
```

13. Trong hàm xử lý sự kiện cbbKTButVe_SelectedIndexChanged, thêm đoạn mã sau:

```
private void cbbKTButVe_SelectedIndexChanged(object sender, EventArgs e)
{
    // Đặt lại độ rộng nét vẽ cho bút vẽ
    butVe.Width = cbbKTButVe.SelectedIndex + 1;
}
```

14. Bổ sung hàm sau đây vào lớp Form1.cs

```
// =====
// Các hàm phụ
// =====

// Hàm ẩn các ô chứa tham số
private void HideParamTextbox()
{
    txtThamSo1.Visible = lblTS1.Visible = false;
    txtThamSo2.Visible = lblTS2.Visible = false;
    txtThamSo3.Visible = lblTS3.Visible = false;
    txtThamSo4.Visible = lblTS4.Visible = false;
    txtThamSo5.Visible = lblTS5.Visible = false;
    txtThamSo6.Visible = lblTS6.Visible = false;
}
```

15. Trong hàm xử lý sự kiện LoaiHinhVe_Click, thêm đoạn mã sau:

```
private void LoaiHinhVe_Click(object sender, EventArgs e)
{
    // Ẩn các ô chứa tham số
    this.HideParamTextbox();

    // Lấy RadioButton được chọn
    RadioButton radHinhVe = sender as RadioButton;

    // Lấy tên của RadioButton được chọn
    string name = radHinhVe.Name;

    // Xét xem thử người dùng muốn vẽ hình gì
    switch (name)
    {
        case "radDoanThang": // Chọn vẽ đoạn thẳng
            break;

        case "radChuNhat": // Chọn vẽ hình chữ nhật
            break;

        case "radEllipse": // Chọn vẽ hình Elip
            break;

        case "radDaGiac": // Chọn vẽ hình đa giác
            break;

        case "radCungTron": // Chọn vẽ cung tròn
            break;

        case "radPie": // Chọn vẽ hình Pie
            break;
    }
    this.Text = name;
}
```

Để vẽ đoạn thẳng, ta cần có tọa độ (x, y) của hai điểm đầu và cuối của đoạn thẳng. Vì thế, ta cần hiển thị 4 ô tham số. Bổ sung đoạn mã sau vào case “radDoanThang”:

```
case "radDoanThang":    // Chọn vẽ đoạn thẳng

    // Hiển thị 4 ô tham số
    txtThamSo1.Visible = lblTS1.Visible = true;
    txtThamSo2.Visible = lblTS2.Visible = true;
    txtThamSo3.Visible = lblTS3.Visible = true;
    txtThamSo4.Visible = lblTS4.Visible = true;

    // Gán lại nhãn cho các label
    lblTS1.Text = "X Đầu";
    lblTS2.Text = "Y Đầu";
    lblTS3.Text = "X Cuối";
    lblTS4.Text = "Y Cuối";

    // Đánh dấu hình muốn vẽ là đoạn thẳng
    kieuHinh = LoaiHinh.DoanThang;

    break;
```

Để vẽ hình chữ nhật và Ellipse, cần phải có tọa độ (x, y) điểm góc trên bên trái và kích thước (dài, rộng) của hình chữ nhật. Vì thế, ta cần hiển thị 4 ô tham số. Bổ sung đoạn mã sau vào case “radChuNhat” và case “radEllipse”:

```
case "radChuNhat":      // Chọn vẽ hình chữ nhật

    // Hiển thị 4 ô tham số
    txtThamSo1.Visible = lblTS1.Visible = true;
    txtThamSo2.Visible = lblTS2.Visible = true;
    txtThamSo3.Visible = lblTS3.Visible = true;
    txtThamSo4.Visible = lblTS4.Visible = true;

    // Gán lại nhãn cho các label
    lblTS1.Text = "Trái";
    lblTS2.Text = "Trên";
    lblTS3.Text = "Dài";
    lblTS4.Text = "Rộng";

    // Đánh dấu hình muốn vẽ là hình chữ nhật
    kieuHinh = LoaiHinh.HinhChuNhat;

    break;
```

```

case "radEllipse":      // Chọn vẽ hình Elip

    // Hiển thị 4 ô tham số
    txtThamSo1.Visible = lblTS1.Visible = true;
    txtThamSo2.Visible = lblTS2.Visible = true;
    txtThamSo3.Visible = lblTS3.Visible = true;
    txtThamSo4.Visible = lblTS4.Visible = true;

    // Gán lại nhãn cho các label
    lblTS1.Text = "Trái";
    lblTS2.Text = "Trên";
    lblTS3.Text = "Dài";
    lblTS4.Text = "Rộng";

    // Đánh dấu hình muốn vẽ là Elip
    kieuHinh = LoaiHinh.Ellipse;

    break;

```

Để vẽ cung tròn hoặc Pie, ta cần tọa độ và kích thước hình chữ nhật chứa cung tròn hay Pie đó. Ngoài ra, cần có thêm góc lệch của cung tròn so với trục hoành và số đo góc của cung hay Pie. Vì thế, ta cần hiển thị 6 ô tham số. Bổ sung đoạn mã sau và case “radCungTron” và case “radPie”:

```

case "radCungTron":     // Chọn vẽ cung tròn

    // Hiển thị 6 ô tham số
    txtThamSo1.Visible = lblTS1.Visible = true;
    txtThamSo2.Visible = lblTS2.Visible = true;
    txtThamSo3.Visible = lblTS3.Visible = true;
    txtThamSo4.Visible = lblTS4.Visible = true;
    txtThamSo5.Visible = lblTS5.Visible = true;
    txtThamSo6.Visible = lblTS6.Visible = true;

    // Gán lại nhãn cho các label
    lblTS1.Text = "Trái";
    lblTS2.Text = "Trên";
    lblTS3.Text = "Dài";
    lblTS4.Text = "Rộng";
    lblTS5.Text = "Góc BD";
    lblTS6.Text = "Số Đo Góc";

    // Đánh dấu hình muốn vẽ là cung tròn
    kieuHinh = LoaiHinh.CungTron;

    break;

```

```
case "radPie":           // Chọn vẽ hình Pie

    // Hiển thị 4 ô tham số
    txtThamSo1.Visible = lblTS1.Visible = true;
    txtThamSo2.Visible = lblTS2.Visible = true;
    txtThamSo3.Visible = lblTS3.Visible = true;
    txtThamSo4.Visible = lblTS4.Visible = true;
    txtThamSo5.Visible = lblTS5.Visible = true;
    txtThamSo6.Visible = lblTS6.Visible = true;

    // Gán lại nhãn cho các label
    lblTS1.Text = "Trái";
    lblTS2.Text = "Trên";
    lblTS3.Text = "Dài";
    lblTS4.Text = "Rộng";
    lblTS5.Text = "Góc BD";
    lblTS6.Text = "Số Đo Góc";

    // Đánh dấu hình muốn vẽ là hình Pie
    kieuHinh = LoaiHinh.Pie;

    break;
```

Để vẽ đa giác, ta cần một tập các điểm. Trong trường hợp này, ta sẽ tạo ra một Form mới để chọn các điểm. Vì thế, không cần các ô tham số.

```
case "radDaGiac":       // Chọn vẽ hình đa giác

    // Đánh dấu hình muốn vẽ là Đa Giác
    kieuHinh = LoaiHinh.DaGiac;

    break;
```

16. Trong lớp Form1.cs, bổ sung các hàm vẽ sau:

```
// Hàm vẽ đoạn thẳng
private void VeDoanThang()
{
    try
    {
        // Lấy tọa độ các điểm
        int xd = int.Parse(txtThamSo1.Text);
        int yd = int.Parse(txtThamSo2.Text);
        int xc = int.Parse(txtThamSo3.Text);
        int yc = int.Parse(txtThamSo4.Text);

        // Vẽ đoạn thẳng
        grp.DrawLine(butVe, xd, yd, xc, yc);
    }
    catch
    {
        // Thông báo lỗi nếu dữ liệu nhập sai
        MessageBox.Show("Dữ liệu nhập không hợp lệ", "Error",
            MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}

// Hàm vẽ hình chữ nhật
private void VeHinhChuNhat()
{
    try
    {
        // Lấy tọa độ điểm bắt đầu vẽ
        int left = int.Parse(txtThamSo1.Text);
        int top = int.Parse(txtThamSo2.Text);

        // Và kích thước của hình chữ nhật
        int height = int.Parse(txtThamSo3.Text);
        int width = int.Parse(txtThamSo4.Text);

        // Nếu Checkbox "Tô màu nền cho hình" được đánh dấu
        // thì tô màu cho hình chữ nhật trước khi vẽ biên
        if (chkMauTo.Checked)
            grp.FillRectangle(coToMau, left, top, width, height);

        // Vẽ đường biên của hình chữ nhật
        grp.DrawRectangle(butVe, left, top, width, height);
    }
    catch
    {
        // Thông báo lỗi nếu dữ liệu nhập sai
        MessageBox.Show("Dữ liệu nhập không hợp lệ", "Error",
            MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}
```

```
// Hàm vẽ hình chữ nhật
private void VeHinhElip()
{
    try
    {
        // Lấy tọa độ điểm bắt đầu HCN chứa Elip
        int left = int.Parse(txtThamSo1.Text);
        int top = int.Parse(txtThamSo2.Text);

        // Và kích thước của hình chữ nhật chứa Elip
        int height = int.Parse(txtThamSo3.Text);
        int width = int.Parse(txtThamSo4.Text);

        // Nếu Checkbox "Tô màu nền cho hình" được đánh dấu
        // thì tô màu cho hình Elip trước khi vẽ biên
        if (chkMauTo.Checked)
            grp.FillEllipse(coToMau, left, top, width, height);

        // Vẽ đường biên của hình Elip
        grp.DrawEllipse(butVe, left, top, width, height);
    }
    catch
    {
        // Thông báo lỗi nếu dữ liệu nhập sai
        MessageBox.Show("Dữ liệu nhập không hợp lệ", "Error",
            MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}

// Hàm vẽ cung tròn
private void VeCungTron()
{
    try
    {
        // Lấy tọa độ điểm bắt đầu HCN chứa cung tròn
        int left = int.Parse(txtThamSo1.Text);
        int top = int.Parse(txtThamSo2.Text);

        // Và kích thước của hình chữ nhật chứa cung tròn
        int height = int.Parse(txtThamSo3.Text);
        int width = int.Parse(txtThamSo4.Text);

        // Lấy góc bắt đầu và số đo góc của cung tròn
        float startAngle = float.Parse(txtThamSo5.Text);
        float sweepAngle = float.Parse(txtThamSo6.Text);

        // Vẽ cung tròn
        grp.DrawArc(butVe, left, top, width, height, startAngle, sweepAngle);
    }
    catch
    {
        // Thông báo lỗi nếu dữ liệu nhập sai
        MessageBox.Show("Dữ liệu nhập không hợp lệ", "Error",
            MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}
```



```
// Hàm vẽ hình Pie
private void VeHinhPie()
{
    try
    {
        // Lấy tọa độ điểm bắt đầu HCN chứa cung tròn
        int left = int.Parse(txtThamSo1.Text);
        int top = int.Parse(txtThamSo2.Text);

        // Và kích thước của hình chữ nhật chứa cung tròn
        int height = int.Parse(txtThamSo3.Text);
        int width = int.Parse(txtThamSo4.Text);

        // Lấy góc bắt đầu và số đo góc của cung tròn
        float startAngle = float.Parse(txtThamSo5.Text);
        float sweepAngle = float.Parse(txtThamSo6.Text);

        // Nếu Checkbox "Tô màu nền cho hình" được đánh dấu
        // thì tô màu cho hình Pie trước khi vẽ biên
        if (chkMauTo.Checked)
            grp.FillPie(coToMau, left, top, width, height, startAngle, sweepAngle);

        // Vẽ đường biên của hình Pie
        grp.DrawPie(butVe, left, top, width, height, startAngle, sweepAngle);
    }
    catch
    {
        // Thông báo lỗi nếu dữ liệu nhập sai
        MessageBox.Show("Dữ liệu nhập không hợp lệ", "Error",
            MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}
```

17. Trong hàm xử lý sự kiện btnVeHinh_Click, thêm đoạn mã sau:

```
private void btnVeHinh_Click(object sender, EventArgs e)
{
    // Xét xem loại hình mà người dùng muốn vẽ là hình gì?
    switch (kiểuHinh)
    {
        case LoaiHinh.DoanThang:
            // Gọi hàm vẽ đoạn thẳng
            this.VeDoanThang();
            break;

        case LoaiHinh.HinhChuNhat:
            // Gọi hàm vẽ hình chữ nhật
            this.VeHinhChuNhat();
            break;

        case LoaiHinh.Ellipse:
            // Gọi hàm vẽ hình Elip
            this.VeHinhElip();
            break;

        case LoaiHinh.DaGiac:
            // Sẽ xử lý sau

            break;

        case LoaiHinh.CungTron:
            // Gọi hàm vẽ cung tròn
            this.VeCungTron();
            break;

        case LoaiHinh.Pie:
            // Gọi hàm vẽ Pie
            this.VeHinhPie();
            break;
    }
}
```

18. Nhấn F5 để chạy chương trình và kiểm tra kết quả

19. Vẽ đa giác: Tạo một lớp mới đặt tên là `HinhDaGiac`. Thêm đoạn mã sau vào file `HinhDaGiac.cs`

```
1 using System;
2 using System.Drawing;
3 using System.Collections.Generic;
4 using System.Text;
5
6 namespace SimpleDrawing
7 {
8     public class HinhDaGiac
9     {
```

```

10     private Point[] dinhdg;
11     private int soDinh;
12
13     // Thuộc tính lưu các đỉnh của đa giác
14     public Point[] CacDinh
15     {
16         get
17         {
18             // Tạo một mảng mới lưu trữ các đỉnh
19             // thực sự của đa giác
20             Point[] dsd = new Point[soDinh + 1];
21
22             // Sao chép các điểm từ dinhdg sang dsd
23             Array.Copy(dinhdg, dsd, soDinh);
24
25             // Đỉnh cuối phải trùng với đỉnh đầu
26             dsd[soDinh] = dinhdg[0];
27
28             return dsd;
29         }
30     }
31
32     // Phương thức tạo lập
33     public HìnhDaGiac()
34     {
35         soDinh = 0;
36         dinhdg = new Point[20];
37     }
38
39     // Hàm thêm một đỉnh vào đa giác
40     public bool ThemDinh(Point p)
41     {
42         // Nếu số đỉnh < 20 thì cho phép thêm
43         if (soDinh < 20)
44         {
45             dinhdg[soDinh++] = p;
46             return true;
47         }
48         else
49             // Ngược lại, không cho thêm vì nhiều đỉnh quá
50             return false;
51     }
52
53     // Hàm xóa danh sách các đỉnh của đa giác
54     public void XoaCacDinh()
55     {
56         soDinh = 0;
57     }
58 }
59

```

20. Tạo một Form mới, đặt tên là Polygon. Thiết lập các thuộc tính của Form này như sau:

Tên thuộc tính	Giá trị
----------------	---------

Text	Polygon: Nhập chuột trái để chọn điểm, nhập phải để vẽ đa giác
Size	450, 450
BackColor	White

21. Nhấp phải lên Form Polygon, chọn View Code rồi thêm đoạn mã sau đây:

```
public partial class Polygon : Form
{
    // =====
    // Khai báo các biến
    // =====

    private HìnhDaGiác dagiac; // Đa giác
    private Pen butVe;         // Bút vẽ
    private Brush coToMau;     // Cọ tô
    private bool toNen;        // Tô nền?

    // =====
    // Các thuộc tính
    // =====

    public Pen ButVe
    {
        set { butVe = value; }
    }

    public Brush CoToMau
    {
        set { coToMau = value; }
    }

    public bool ToNen
    {
        set { toNen = value; }
    }
}
```

22. Trong phương thức tạo lập của Form Polygon, khởi tạo giá trị cho các biến như sau:

```
public Polygon()
{
    InitializeComponent();

    // Khởi tạo giá trị mặc định cho Pen, Brush
    butVe = Pens.Blue;
    coToMau = Brushes.YellowGreen;

    // Mặc định, ta không tô nền
    toNen = true;
}
```

23. Thêm hàm xử lý sự kiện Load cho Form Polygon rồi bổ sung đoạn mã sau

```
private void Polygon_Load(object sender, EventArgs e)
{
    // Khởi tạo một đa giác rỗng
    dagiac = new HìnhDaGiac();
}
```

24. Thêm hàm xử lý sự kiện MouseDown cho Form Polygon như sau

```
private void Polygon_MouseDown(object sender, MouseEventArgs e)
{
    // Lấy công cụ vẽ từ Form
    Graphics g = this.CreateGraphics();

    // Nếu nhấn chuột trái thì thêm đỉnh
    if (e.Button == MouseButtons.Left)
    {
        // Nếu số lượng đỉnh > 20 thì không cho thêm nữa
        if (dagiac.ThemDinh(e.Location) == false)
            MessageBox.Show("Bạn chỉ được phép tạo đa giác tối đa 20 đỉnh.
        else
        {
            // Ngược lại, vẽ điểm đã đánh dấu lên màn hình
            g.DrawRectangle(new Pen(Color.Red, 2.Of), e.X, e.Y, 2, 2);
        }
    }
    else
    {
        // Lấy các đỉnh của đa giác
        Point[] dinh = dagiac.CacDinh;

        // Nếu đa giác có 3 đỉnh trở lên thì mới vẽ
        if (dinh.Length > 3)
        {
            if (toNen)
                // Tô màu nền cho đa giác
                g.FillPolygon(coToMau, dinh);

            // Vẽ biên đa giác
            g.DrawPolygon(butVe, dinh);

            // Xóa các đỉnh của đa giác để vẽ đa giác mới
            dagiac.XoaCacDinh();
        }
        else
            MessageBox.Show("Đa giác phải có từ 3 đỉnh trở lên", "Error");
    }
}
```

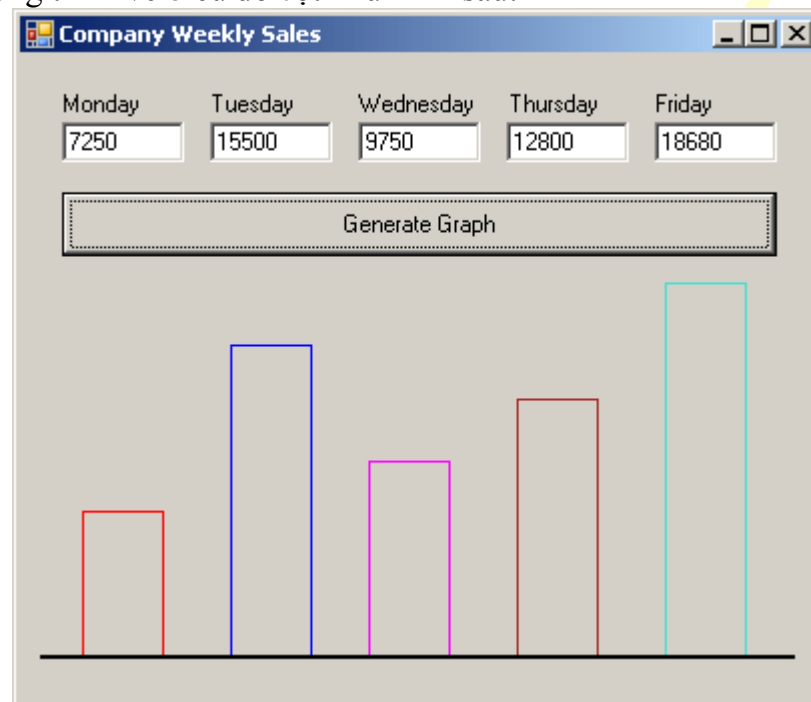
25. Quay trở lại Form1.cs. Trong hàm xử lý sự kiện Click cho nút Vẽ Hình, bổ sung đoạn code sau vào mục *case LoaiHinh.DaGiac* của lệnh switch.

```
case LoaiHinh.DaGiac:  
    // Hiển thị Form Polygon  
    Polygon p = new Polygon();  
    p.ButVe = butVe;  
    p.CoToMau = coToMau;  
    p.ToNen = chkMauTo.Checked;  
    p.Show();  
    break;
```

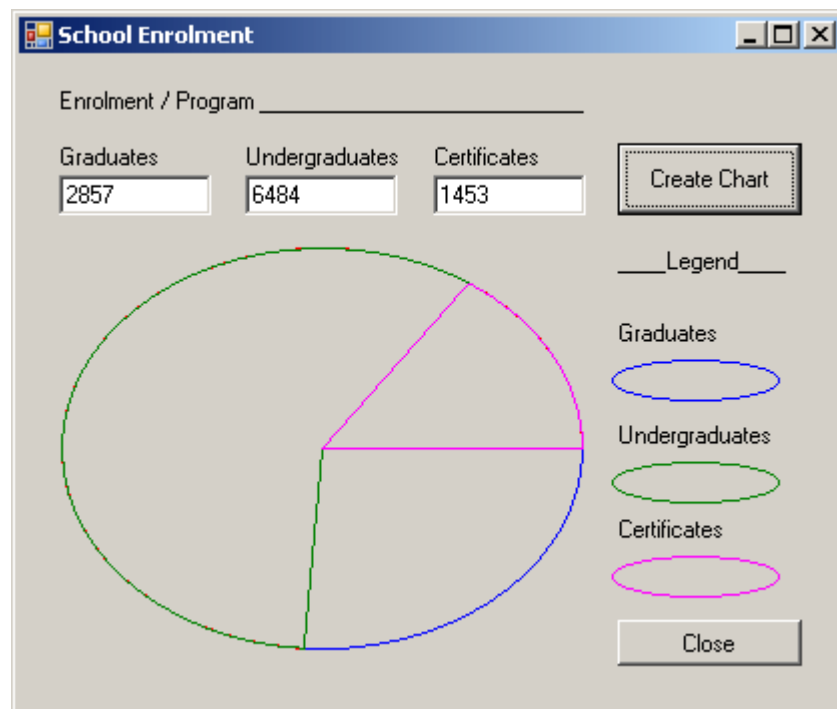
26. Nhấn F5 để chạy chương trình và kiểm tra kết quả.

IV. Bài tập làm thêm

- Viết chương trình vẽ biểu đồ cột như hình sau:



- Viết chương trình vẽ biểu đồ tròn như hình sau:



3. Cải tiến chương trình vẽ trên để có thể lưu các hình đã vẽ ra file (*.jpg, *.bmp,...)
4. Cải tiến chương trình vẽ trên để hình ảnh không bị xóa khi phóng to, thu nhỏ cửa sổ hay bị một cửa sổ khác che lấp.

Gợi ý:

- Sử dụng lớp Bitmap để lưu các hình đã vẽ (không vẽ trực tiếp lên Panel)
- Xử lý sự kiện Paint của Panel để vẽ lại Bitmap.
- Dùng phương thức Save(string fileName) của lớp Bitmap để lưu hình.