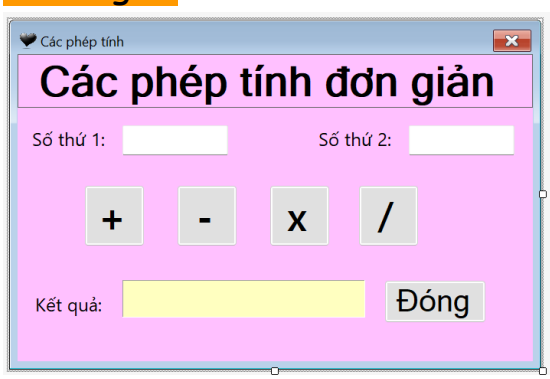
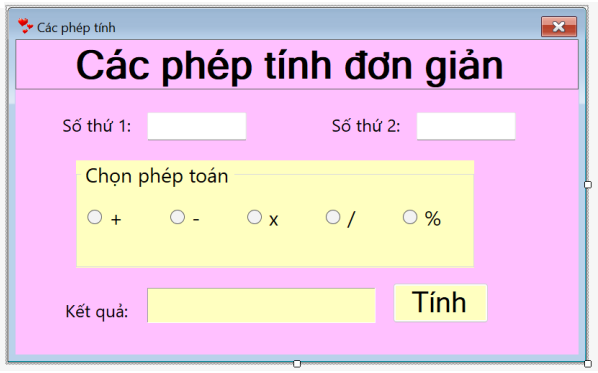
Trần Đặng Mỹ Tiên - 2151050455

Chương 2





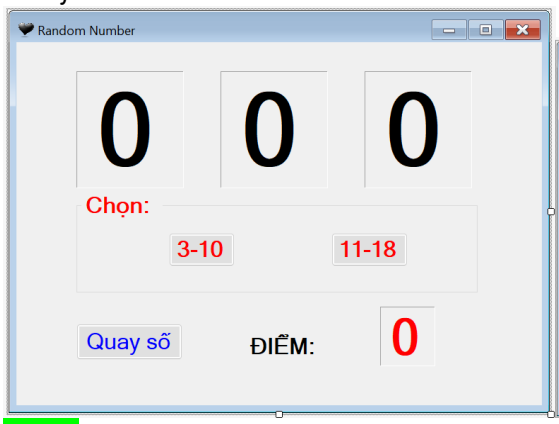
Bài 1/

```
 private void bntCong_Click(object sender, EventArgs e)
      {
         int a = int.Parse(txtSo1.Text);
         int b  = int.Parse(txtSo2.Text);
         int kq = a + b;
         lbKQ.Text = kq.ToString();
      }
private void bntTru_Click(object sender, EventArgs e)
      {
         int a = int.Parse(txtSo1.Text);
         int b = int.Parse(txtSo2.Text);
         int kq = a - b;
         lbKQ.Text = kq.ToString();
      }
private void bntDong_Click(object sender, EventArgs e)
      {
         DialogResult rel = MessageBox.Show("Bạn có chắc chắn muốn thoát khỏi form?", "Thông báo",
MessageBoxButtons.OKCancel);
         if (rel == DialogResult.OK)
            this.Close();
      }
private void bntChia_Click(object sender, EventArgs e)
      {
         int a = int.Parse(txtSo1.Text);
         int b = int.Parse(txtSo2.Text);
         if (b != 0) //Nếu số bị chia khác 0 mới thực hiện phép tính
            lbKQ.Text = String.Format("{0:0.00}", (double)a / b);
         else
            lbKQ.Text = "Số bị chia phải khác 0";
      }
 private void bntNhan_Click(object sender, EventArgs e)
      {
         int a = int.Parse(txtSo1.Text);
         int b = int.Parse(txtSo2.Text);
         int kq = a * b;
         lbKQ.Text = kq.ToString();
      }
```

Bài 2/

```
 private void bntTinh_Click(object sender, EventArgs e)
      {
```

```
        try
        {
            int a = checked(int.Parse(txtSo1.Text));
            int b = checked(int.Parse(txtSo2.Text));
            if (rdCong.Checked) //Kiểm tra rd nào được chọn
                lbKQ.Text = string.Format("{0}", checked(a + b)); //Kiểm tra tràn số
            else if (rdTru.Checked)
                lbKQ.Text = string.Format("{0}", a - b);
            else if (rdNhan.Checked)
                lbKQ.Text = string.Format("{0}", checked(a * b)); //Kiểm tra tràn số
            else
            {
                if (b == 0) //Lỗi mặc đặc định DivideByZeroException thì nó xuất câu tiếng anh => Xuất tiếng
việt thì tự ném lỗi
                    throw new DivideByZeroException("Số chia phải khác 0");
                if (rdChia.Checked)
                    lbKQ.Text = String.Format("{0:0.00}", (double) a / b);
                else
                    lbKQ.Text = String.Format("{0}", a / b);
            }
        }
        catch (FormatException)
        {
            lbKQ.Text = "Bạn phải nhập hai số";
        }
        catch (OverflowException)
        {
            lbKQ.Text = "Số quá lớn";
        }
        catch (DivideByZeroException ex)
        {
            lbKQ.Text = ex.Message;
        }
    }
```



Bài 3/

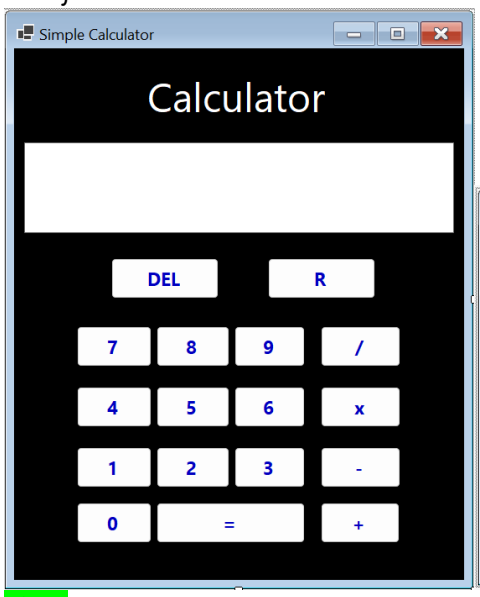
```
Random r = new Random();
    int mark = 0;
private void bntQuaySo_Click(object sender, EventArgs e)
    {
        int so1, so2, so3;
        so1 = r.Next(1, 7);
        so2 = r.Next(1, 7);
        so3 = r.Next(1, 7);
```

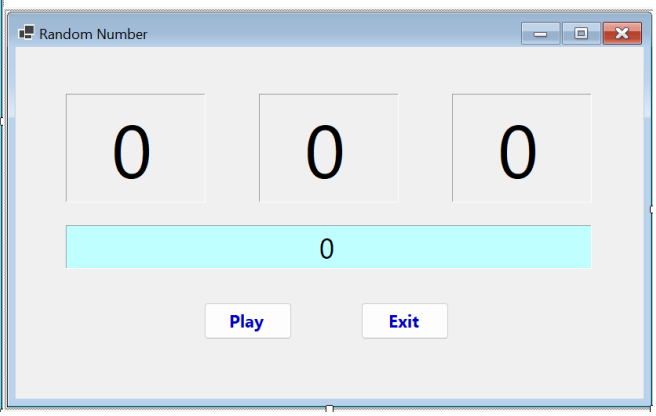
```
            lb1.Text = so1.ToString();
            lb2.Text = so2.ToString();
            lb3.Text = so3.ToString();

            if (rd3.Checked)
               if (so1 + so2 + so3 <= 10)
                  mark += 10;
               else
                  mark -= 10;
            else
               if (so1 + so2 + so3 > 10)
                  mark += 10;
               else
                  mark -= 10;

            lbDiem.Text = mark.ToString();
         }
```

Bài 4/

```
 Random r = new Random();
         string[] toantu = { "+", "-", "X", "/" };
         int index = 0;
//Tạo sự kiện nhấn cho các số 0-9 và dấu -
         private void bnt8_Click(object sender, EventArgs e)
         {
            Button bt = (Button)sender;
            txtToan.Text += bt.Text;

         }
private void bntXem_Click(object sender, EventArgs e)
         {
            int kq = 0;
            switch (index)
            {
               case 0:
                  kq = int.Parse(lbSo1.Text) + int.Parse(lbSo2.Text);
                  break;
               case 1:
                  kq = int.Parse(lbSo1.Text) - int.Parse(lbSo2.Text);
                  break;
               case 2:
                  kq = int.Parse(lbSo1.Text) * int.Parse(lbSo2.Text);
                  break;
               case 3:
                  kq = int.Parse(lbSo1.Text) / int.Parse(lbSo2.Text);
                  break;

            }
            try
            {
               int user = int.Parse(txtToan.Text);
               if (user == kq)
                  lbKQ.Text = "Chúc mừng bạn! Chính xác";
               else
                  lbKQ.Text = "Tiếc quá! Kết quả đúng là " + kq.ToString();
```

```
        }
        catch (FormatException)
        {
            MessageBox.Show("Bạn phải nhập kết quả dự đoán là số!", "Thông báo");
        }
    }
private void bntTroVe_Click(object sender, EventArgs e)
    {
        if (txtToan.Text == " ") return;
        txtToan.Text = txtToan.Text.Substring(0, txtToan.TextLength - 1);
    }
private void bntTru_Click_1(object sender, EventArgs e)
    {
         if (txtToan.Text != " ") return;
         txtToan.Text = "-";
    }
```





Bài 5

```
//Tạo sự kiện nhấn cho các số
    private void bt0_Click(object sender, EventArgs e)
    {
        Button bt = (Button)sender;
        lbTinh.Text += bt.Text;
    }

    //Tạo sự kiện nhấn cho các operation (dấu)
    //Tạo sự kiện nhấn để hiển thị số hoặc dấu lên lable cho người dùng thấy
    private void Operation_Click(object sender, EventArgs e)
    {
        Button bt = (Button)sender;
        lbTinh.Text += bt.Text;
    }

    //Xóa 1
    private void btR_Click(object sender, EventArgs e)
    {
        if (lbTinh.Text == "") return; //Khi lable rỗng thì không có gì để xóa nữa nên retrun
        //Trường hợp của else
```

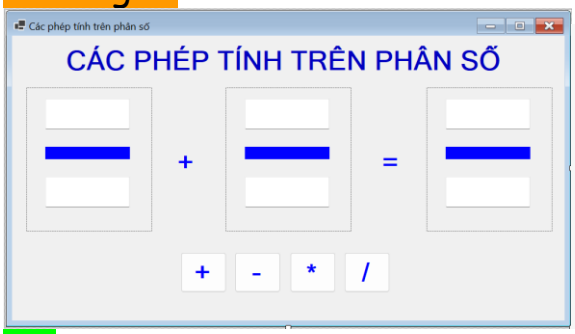
```
        lbTinh.Text = lbTinh.Text.Substring(0, lbTinh.Text.Length - 1); //Cắt chuỗi từ vị trí 0 đến độ dài trừ đi 1 đơn vị

    }
    //Xóa hết
    private void btDel_Click(object sender, EventArgs e)
    {
        if (lbTinh.Text == "") return; //Khi lable rỗng thì không có gì để xóa nữa nên retrun
        //Trường hợp else
        lbTinh.Text = ""; //Gán chuỗi bằng rỗng
    }

    private void btBang_Click(object sender, EventArgs e)
    {
        try
        {
            int pos = 0;
            char dau = '+';
            try
            {
                if (lbTinh.Text.Contains("+")) //Contains: Có chứa kí tự '+'
                    pos = lbTinh.Text.IndexOf('+'); //Indexof : trả về chỉ số vị trí xuất hiện lần đầu tiên một ký tự chỉ định hoặc một chuỗi con chỉ định trong chuỗi
                else if (lbTinh.Text.Contains("-"))
                {
                    pos = lbTinh.Text.IndexOf('-');
                    dau = '-';
                }
                else if (lbTinh.Text.Contains("x"))
                {
                    pos = lbTinh.Text.IndexOf('x');
                    dau = 'x';
                }
                else
                {
                    pos = lbTinh.Text.IndexOf('/');
                    dau = '/';
                }
                int a = int.Parse(lbTinh.Text.Substring(0, pos)); //Parse: ép kiểu về số nguyên
                int b = int.Parse(lbTinh.Text.Substring(pos + 1));
                switch(dau)
                {
                    case '+': lbTinh.Text += " = " + (a + b).ToString(); break;
                    case '-': lbTinh.Text += " = " + (a - b).ToString(); break;
                    case 'x': lbTinh.Text += " = " + (a * b).ToString(); break;
                    case '/': lbTinh.Text += " = " + (a / b).ToString(); break;
                    default: break;
                }
            }
            catch (OverflowException)
            {
                MessageBox.Show("Vượt quá dữ liệu", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Error);
                lbTinh.Text = "";
            }
```

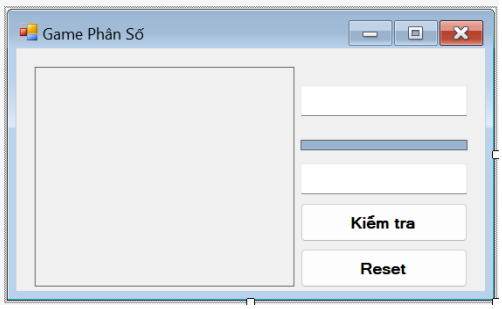
```
            catch (DivideByZeroException)
            {

                MessageBox.Show("Lỗi chia cho 0", "Thông báo", MessageBoxButtons.OK,
MessageBoxIcon.Error);
                lbTinh.Text = "";
            }
        }
        catch (Exception) { }
```

Bài 6/

```
 Random rand = new Random();
      int Diem = 0;
      public Form1()
      {
         InitializeComponent();
      }

      private void btPlay_Click(object sender, EventArgs e)
      {
         int so1 = rand.Next(1, 7); //Random từ 1 - 6
         int so2 = rand.Next(1, 7);
         int so3 = rand.Next(1, 7);
         lbSo1.Text = so1.ToString();
         lbSo2.Text = so2.ToString();
         lbSo3.Text = so3.ToString();
         if (so1 == so2 && so1 == so3)
         {
            Diem += 100;
         }
         else
            Diem -= 10;
         lbDiem.Text = Diem.ToString();

      }

      private void btExit_Click(object sender, EventArgs e)
      {
         Close();
      }
```

Chương 3





Bài 1

```
private void bntCong_Click(object sender, EventArgs e)
      {
         lbDau.Text = bntCong.Text;
         PhanSo ps1, ps2;
```

```
    try
    {
        ps1 = new PhanSo(int.Parse(txtTu1.Text), int.Parse(txtMau1.Text));
        ps2 = new PhanSo(int.Parse(txtTu2.Text), int.Parse(txtMau2.Text));
        PhanSo kq = ps1.cong(ps2);
        txtTu.Text = kq.TS.ToString();
        txtMau.Text = kq.MS.ToString();
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message, "Error!");
    }
}

private void bntTru_Click(object sender, EventArgs e)
{
    lbDau.Text = bntTru.Text;
    PhanSo ps1, ps2;
    try
    {
        ps1 = new PhanSo(int.Parse(txtTu1.Text), int.Parse(txtMau1.Text));
        ps2 = new PhanSo(int.Parse(txtTu2.Text), int.Parse(txtMau2.Text));
        PhanSo kq = ps1.tru(ps2);
        txtTu.Text = kq.TS.ToString();
        txtMau.Text = kq.MS.ToString();
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message, "Error!");
    }
}

private void bntNhan_Click(object sender, EventArgs e)
{
    lbDau.Text = bntNhan.Text;
    PhanSo ps1, ps2;
    try
    {
        ps1 = new PhanSo(int.Parse(txtTu1.Text), int.Parse(txtMau1.Text));
        ps2 = new PhanSo(int.Parse(txtTu2.Text), int.Parse(txtMau2.Text));
        PhanSo kq = ps1.nhan(ps2);
        txtTu.Text = kq.TS.ToString();
        txtMau.Text = kq.MS.ToString();
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message, "Error!");
    }
}

private void bntChia_Click(object sender, EventArgs e)
{
    lbDau.Text = bntChia.Text;
    PhanSo ps1, ps2;
    try
```

```
        {
            ps1 = new PhanSo(int.Parse(txtTu1.Text), int.Parse(txtMau1.Text));
            ps2 = new PhanSo(int.Parse(txtTu2.Text), int.Parse(txtMau2.Text));
            PhanSo kq = ps1.chia(ps2);
            txtTu.Text = kq.TS.ToString();
            txtMau.Text = kq.MS.ToString();
        }
        catch (Exception ex)
        {
            MessageBox.Show(ex.Message, "Error!");
        }
    }
```

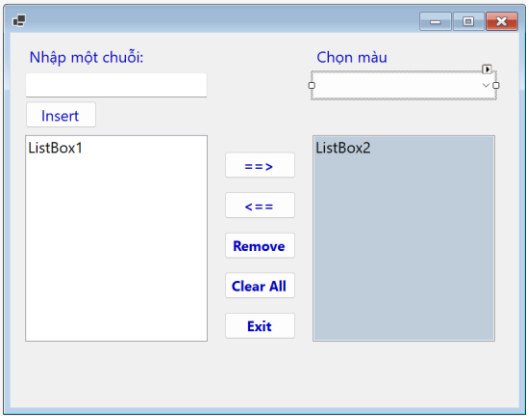
Bài 2.

```
CLASS MAIN
private Random random = new Random();
private Fraction a;
private Fraction b;
private string operatorString;
private Fraction result;
private void GenerateRandomProblem()
    {
        int num1 = random.Next(1, 10); //tử
        int num2 = random.Next(1, 10);
        int den1 = random.Next(1, 10); //tử
        int den2 = random.Next(1, 10);

        a = new Fraction(num1, den1); //Tạo phân số nhờ
        b = new Fraction(num2, den2);
        int operatorIndex = random.Next(1,5);
        switch (operatorIndex)
        {
            case 1:
                operatorString = "+";
                result = Fraction.Add(a, b);
                break;
            case 2:
                operatorString = "-";
                result = Fraction.Subtract(a, b);
                break;
            case 3:
                operatorString = "*";
                result = Fraction.Multiply(a, b);
                break;
            case 4:
                operatorString = "/";
                result = Fraction.Divide(a, b);
                break;
        }
        //$ Để chèn biểu thức
        labelProblem.Text = $"{a.Numerator}/{a.Denominator} {operatorString}
{b.Numerator}/{b.Denominator} = ?";
        textBoxNumerator.Clear(); //Xóa cái tử dự đoán khi phát sinh phép tính 2 phân số mới
        textBoxDenominator.Clear(); //Xóa cái mẫu dự đóan khi phát sinh phép tính 2 phân số mới
        textBoxNumerator.Focus(); //Đặt con trỏ vào textbox đó
    }
```

```
      //Kiểm tra
      private void button1_Click(object sender, EventArgs e)
      {
         try
         {   //Tạo một phân số do người dùng nhập (để dự đoán kết quả 2 phép tính)
            Fraction userAnswer = new Fraction(int.Parse(textBoxNumerator.Text),
int.Parse(textBoxDenominator.Text));
            if (userAnswer.Numerator == result.Numerator && userAnswer.Denominator ==
result.Denominator)
            {
               MessageBox.Show("Bạn đã trả lời đúng!");
               GenerateRandomProblem();
            }
            else
            {
               MessageBox.Show("Bạn đã trả lời sai!");
               ResetMouseEventArgs();
            }
         }
         catch (FormatException)
         {
           MessageBox.Show("Vui lòng nhập đáp án ở dạng phân số!");
         }
      }

      //Tạo lại
      private void button2_Click(object sender, EventArgs e)
      {

         labelProblem.Text = "";
         textBoxNumerator.Text = "";
         textBoxDenominator.Text = "";
         GenerateRandomProblem();

      }
CLASS PHỤ
private int numerator; // tử số
      private int denominator; // mẫu số

      //Tọa phương thức khởi tạo có tham số rồi nên hàm constructor sẽ không được tạo nữa (không cần
dùng trong bài này)
      public Fraction(int numerator, int denominator) //Phương thức khởi tạo có tham số
      {
         this.numerator = numerator;
         this.denominator = denominator;
      }

      public static Fraction Add(Fraction a, Fraction b) //Phương thức cộng 2 phân số
      {
         int numerator = a.numerator * b.denominator + b.numerator * a.denominator;
         int denominator = a.denominator * b.denominator;
         return new Fraction(numerator, denominator);
      }
```

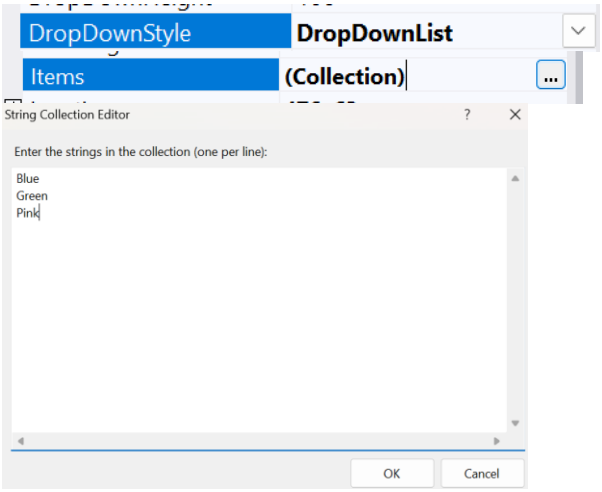
```
public static Fraction Subtract(Fraction a, Fraction b) //Phương thức trừ 2 phân số
{
    int numerator = a.numerator * b.denominator - b.numerator * a.denominator;
    int denominator = a.denominator * b.denominator;
    return new Fraction(numerator, denominator);
}

public static Fraction Multiply(Fraction a, Fraction b) //Phương thức nhân 2 phân số
{
    int numerator = a.numerator * b.numerator;
    int denominator = a.denominator * b.denominator;
    return new Fraction(numerator, denominator);
}

public static Fraction Divide(Fraction a, Fraction b) //Phương thức chia 2 phân số
{
    int numerator = a.numerator * b.denominator;
    int denominator = a.denominator * b.numerator;
    return new Fraction(numerator, denominator);
}

// getters và setters
public int Numerator
{
    get { return numerator; }
    set { numerator = value; }
}

public int Denominator
{
    get { return denominator; }
    set { denominator = value; }
}
```

Chương 4



- Nhập một chuỗi trên textbox, click nút Insert, chuỗi sẽ được thêm vào listbox bên trái, textbox được xóa rỗng.
- Chọn một hoặc nhiều phần tử từ listbox bên trái, click nút ➔, các phần tử được chọn sẽ chuyển sang listbox bên phải.
- Chỉ được chọn một phần tử từ listbox bên phải, click nút ➔, phần tử được chọn sẽ chuyển sang listbox bên trái.
- Click nút Remove: các phần tử được chọn trên hai listbox sẽ bị xóa.
- Click nút Clear All: xóa tất cả các phần tử trên hai listbox.
- Click nút Exit: đóng ứng dụng.
- Chọn một tên màu trên danh sách, màu nền của listbox bên phải sẽ thay đổi tương ứng.

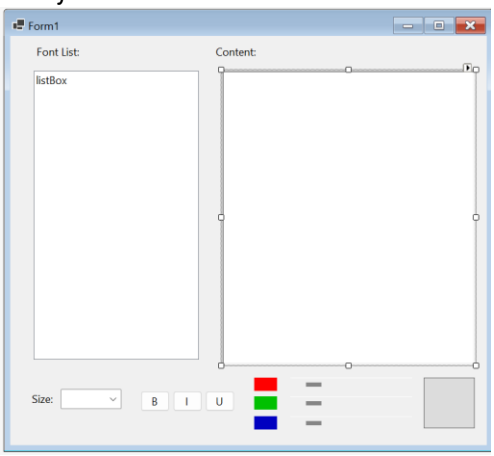
ComboBox




```
private void btInsert_Click(object sender, EventArgs e)
    {
        try
        {
            if (txtNhap.Text != "")
            {
                ListBox1.Items.Add(txtNhap.Text);
                txtNhap.Clear();
                txtNhap.Focus();
            }
        }
        catch(FormatException)
        {
            MessageBox.Show("Vui lòng nhập dữ liệu!!", "Thông báo", MessageBoxButtons.OK,
MessageBoxIcon.Error);
            txtNhap.Clear();
            txtNhap.Focus();
        }
        catch(OverflowException)
        {
            MessageBox.Show("Vượt quá dữ liệu!!", "Thông báo", MessageBoxButtons.OK,
MessageBoxIcon.Error);
            txtNhap.Clear();
            txtNhap.Focus();
        }
    }
bool CheckListBox(ListBox tmp)
    {
```

```
        if (tmp.Items.Count > 0)
            return true;
        return false;
    }

    private void btRight_Click(object sender, EventArgs e)
    {
      try
        {
            ListBox1.SelectionMode = SelectionMode.MultiSimple;//cho phép chọn nhiều item
            for (int i = ListBox1.SelectedItems.Count - 1; i >= 0; i--)
                ListBox2.Items.Add(ListBox1.Items[i]);
            for (int i = ListBox1.SelectedItems.Count - 1; i >= 0; i--)
                ListBox1.Items.Remove(ListBox1.Items[i]);
        }
        catch(Exception) { }
    }
    private void btLeft_Click(object sender, EventArgs e)
    {
        try
        {
            ListBox1.Items.Add(ListBox2.SelectedItem);
            ListBox2.Items.Remove(ListBox2.SelectedItem);
        }
        catch (Exception) { }
    }

    private void btRemove_Click(object sender, EventArgs e)
    {
        if (CheckListBox(ListBox1) || CheckListBox(ListBox2))
        {
            for (int i = ListBox2.SelectedItems.Count - 1; i >= 0; i--)
            {
                ListBox2.Items.Remove(ListBox2.Items[i]);
            }
            for (int i = ListBox1.SelectedItems.Count - 1; i >= 0; i--)
            {
                ListBox1.Items.Remove(ListBox1.Items[i]);
            }
        }
        else
        {
            MessageBox.Show("List trống, không xóa được!!", "Thông báo", MessageBoxButtons.OKCancel,
MessageBoxIcon.Error);
        }
    }

    private void btClear_Click(object sender, EventArgs e)
    {
        if (ListBox1.Items.Count > 0)
            ListBox1.Items.Clear();
        if (ListBox2.Items.Count > 0)
            ListBox2.Items.Clear();
    }
```

```
    private void btExit_Click(object sender, EventArgs e)
    {
        DialogResult ex = MessageBox.Show("Bạn có chắc chắn muốn thoát không?", "Thông báo",
MessageBoxButtons.OKCancel, MessageBoxIcon.Question);
        if (ex == DialogResult.OK)
            this.Close();
    }

    private void comboBoxMau_SelectedIndexChanged(object sender, EventArgs e)
    {
        if (comboBoxMau.SelectedIndex == 0)
            ListBox2.BackColor = Color.Blue;
        else if (comboBoxMau.SelectedIndex == 1)
            ListBox2.BackColor = Color.Green;
        else
            ListBox2.BackColor = Color.Pink;
    }
```



Yêu cầu:

- Chọn một đoạn văn bản, chọn font chữ, size, B, /. U hoặc rê các thanh cuộn màu để áp dụng cho phần văn bản đã chọn.

Bên trái là ListBox bên phải là RichTextBox

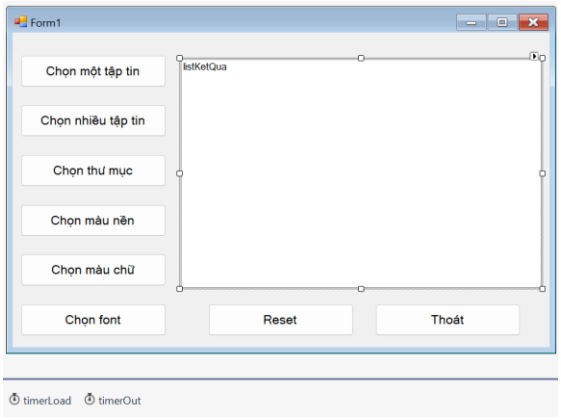
Thanh trượt là : HScrollBar

```
bool isChecked = false;
private void Form1_Load(object sender, EventArgs e)
    {
        InstalledFontCollection fonts = new InstalledFontCollection();
        foreach (FontFamily family in fonts.Families)
            listBox.Items.Add(family.Name); //Add font vao listBox
        FontSize(); //Hàm tạo size chữ
    }

    private void FontSize()//size chữ
    {
        for (int i = 8; i <= 32; i += 2)
            comboBox.Items.Add(i.ToString());
        comboBox.SelectedIndex = 3; //size chữ mặc định là = 14)
    }
    private void listBox_SelectedIndexChanged(object sender, EventArgs e)
    {
        string fontName = listBox.SelectedItem.ToString(); // lấy font chữ trong listbox
        //lấy văn bản được chọn trong richbox
        int selectionStart = richTextBox.SelectionStart;
```

```
            int selectionLength = richTextBox.SelectionLength;
            //đặt font chữ mới cho văn bản đc chọn
            richTextBox.SelectionFont = new Font(fontName, richTextBox.SelectionFont.Size,
richTextBox.SelectionFont.Style);
            //khôi phục vị trí chọn ban đầu
            richTextBox.Select(selectionStart, selectionLength);
        }

        private void comboBox_SelectedIndexChanged(object sender, EventArgs e)
        {
            float fontSize = float.Parse(comboBox.SelectedItem.ToString());
            //lấy văn bản đc chọn trong richtextbox
            int selectionStart = richTextBox.SelectionStart;
            int selectionLength = richTextBox.SelectionLength;
            //đặt kích thước chữ cho văn bản được chọn
            richTextBox.SelectionFont = new Font(richTextBox.Font.FontFamily, fontSize,
richTextBox.Font.Style);
            //khôi phục vị trí ban đầu
            richTextBox.Select(selectionStart, selectionLength);
        }

        private void rdB_CheckedChanged(object sender, EventArgs e)
        {
            isChecked = rdB.Checked;
        }

        private void rdI_CheckedChanged(object sender, EventArgs e)
        {
            isChecked = rdI.Checked;
        }

        private void rdU_CheckedChanged(object sender, EventArgs e)
        {
            isChecked = rdU.Checked;
        }

        private void rdB_Click(object sender, EventArgs e)
        {
            FontStyle style = richTextBox.SelectionFont.Style;
            if(rdB.Checked && !isChecked)
            {
                rdB.Checked = false;
                style &= FontStyle.Bold;
            }
            else
            {
                rdB.Checked = true;
                isChecked = false;
                style |= FontStyle.Bold;
            }
            richTextBox.SelectionFont = new Font(richTextBox.SelectionFont, style);
        }

        private void rdI_Click(object sender, EventArgs e)
        {
```

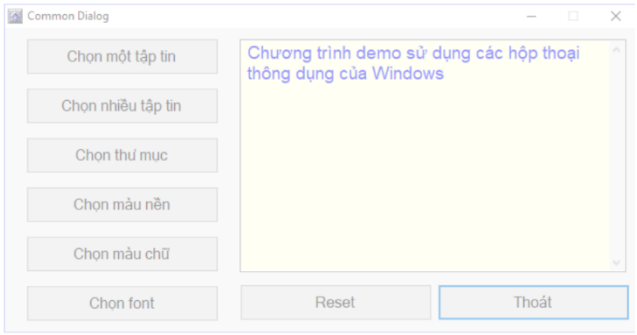
```
    FontStyle style = richTextBox.SelectionFont.Style;
    if(rdB.Checked && !isChecked)
    {
        rdB.Checked = false;
        style |= FontStyle.Italic;
    }
    {
        rdB.Checked = true;
        isChecked = false;
        style |= FontStyle.Italic;
    }
    richTextBox.SelectionFont = new Font(richTextBox.SelectionFont, style);

}

private void rdU_Click(object sender, EventArgs e)
{
    FontStyle style = richTextBox.SelectionFont.Style;
    if(rdB.Checked && !isChecked)
    {
        rdB.Checked = false;
        style &= FontStyle.Underline;
    }
    else
    {
        rdB.Checked = true;
        isChecked = false;
        style |= FontStyle.Underline;
    }
    richTextBox.SelectionFont = new Font(richTextBox.SelectionFont, style);
}

private void hScrollBar1_Scroll(object sender, ScrollEventArgs e)
{
    //lấy giá trị màu từ hscrollbar
    Color color = Color.FromArgb(hScrollBar1.Value, hScrollBar2.Value, hScrollBar3.Value);
    //lấy văn bản đc chọn trong richtextbox
    int selectionStart = richTextBox.SelectionStart;
    int selectionLength = richTextBox.SelectionLength;
    //đặt màu cho văn bản dc chọn
    richTextBox.SelectionColor = color;
    //khoi phuc vi tri ban dau
    richTextBox.Select(selectionStart, selectionLength);

}

private void hScrollBar2_Scroll(object sender, ScrollEventArgs e)
{
    Color color = Color.FromArgb(hScrollBar1.Value, hScrollBar2.Value, hScrollBar3.Value);
    int selectionStart = richTextBox.SelectionStart;
    int selectionLength = richTextBox.SelectionLength;
    richTextBox.SelectionColor = color;
    richTextBox.Select(selectionStart, selectionLength);
}
```

```
private void hScrollBar3_Scroll(object sender, ScrollEventArgs e)
{
    Color color = Color.FromArgb(hScrollBar1.Value, hScrollBar2.Value, hScrollBar3.Value);
    int selectionStart = richTextBox.SelectionStart;
    int selectionLength = richTextBox.SelectionLength;
    richTextBox.SelectionColor = color;
    richTextBox.Select(selectionStart, selectionLength);
}
```

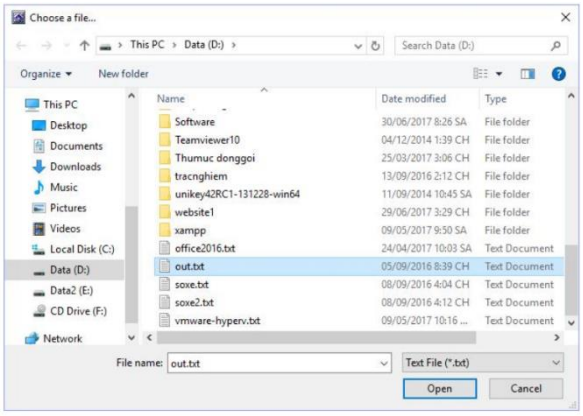


Yêu cầu:

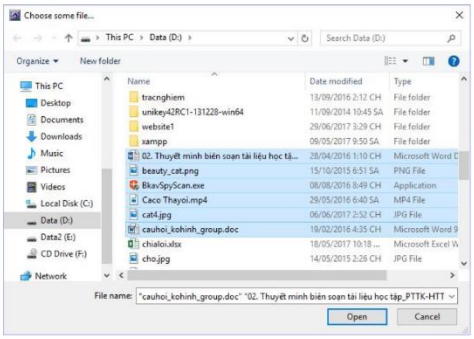
- Khi chương trình thực thi, form từ trong suốt chuyển động rõ dần cho đến khi rõ hoàn toàn.
- Khi đóng chương trình, form sẽ không đóng ngay mà sẽ mờ dần cho đến khi đóng hoàn toàn.

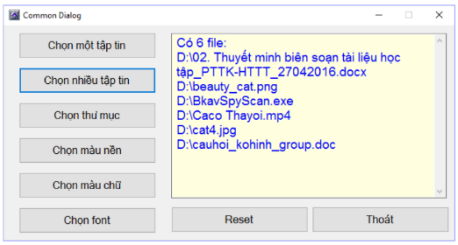


- Chọn một tập tin: mở hộp thoại OpenFileDialog, sau khi chọn, tên tập tin hiển thị trên textbox.

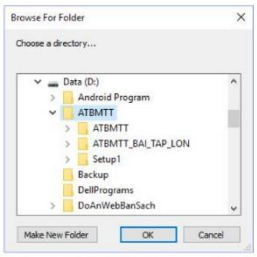


- Chọn nhiều tập tin: mở hộp thoại OpenFileDialog, cho phép chọn nhiều tập tin, sau khi chọn, danh sách tên các tập tin hiển thị trên textbox



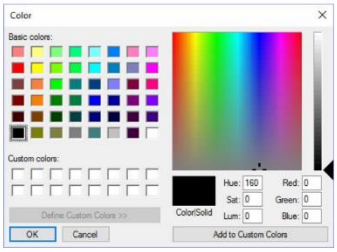


- Chọn thư mục: mở hộp thoại FolderBrowseDialog, sau khi chọn, tên thư mục được chọn hiển thị trên textbox

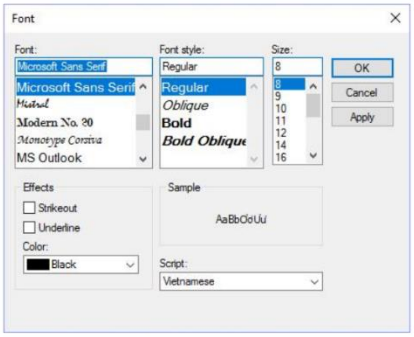


- Chọn màu nền: mở hộp thoại ColorDialog, dạng bình thường, sau khi chọn màu, màu nền textbox sẽ thay đổi tương ứng

- Chọn màu chữ: mở hộp thoại ColorDialog, sau khi chọn màu, màu chữ trên textbox sẽ thay đổi tương ứng.



- Chọn Font: mở hộp thoại FontDialog, sau khi chọn font, văn bản trên textbox sẽ thay đổi font chữ tương ứng.



Reset: trả lại trạng thái ban đầu

Thoát: đóng ứng dụng, có hiển thị cảnh báo đóng form

```
//Khi khai báo biến kiểu bool mà không gán giá trị cho nó, mặc định nó sẽ nhận giá trị false.
    private bool aOpen;
    private bool aClose;
    public Form1()
    {
        this.Opacity = 0; //Khi form mở thì nó ở trạng thái mờ (độ rox bằng 0)
        InitializeComponent();
    }
    //Thêm 1 tập tin
    private void btnMTT_Click(object sender, EventArgs e)
    {
        OpenFileDialog dlg = new OpenFileDialog();
        //Đặt tên tiêu đề cho dlg
        //dgl.Title = "Hộp thoại mở file"!
        //Lọc file chỉ định
        //dgl.Filter = "File ảnh|*.jpg;*.jpeg;*.png;*.gif;*.tif|ALL File|*.*";
        dlg.Filter = "txt files (*.txt)|*.txt|All files (*.*)|*.*";
        if (dlg.ShowDialog() == DialogResult.OK) //Kiểm tra người dùng đã chọn
        {  //Lấy tên thì là SaveFileName
            listKetQua.Items.Add(dlg.FileName); //Add đường dẫn
            //Lấy ảnh
```

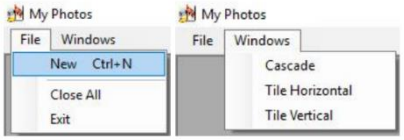
```
        //pictureBox1.Image = Image.FromFile(f.FileName);
    }

}
//MỞ RỘNG LƯU FILE (SAVEFILEDIALOG) (RichtextBox : Cái cho người dùng nhập nhiều, chèn ảnh...)
//SaveFileDialog f = new SaveFileDialog();
////Đặt tên tiêu đề
//f.Title = "Hộp thoại lưu file"!
////Định dạng nơi muốn lưu
//f.Filter = "txt files (*.txt)|*.txt|All files (*.*)|*.*";
//if (f.ShowDialog() == DialogResult.OK)
//{
//    File.WriteAllText(f.FileName, txtTuThem.Text(cái người dùng muốn lưu));
//}
//else
//{
//    MessageBox.Show("Bạn chưa lưu nội dung");
//}




//Thêm nhiều tập tin
private void btnNTT_Click(object sender, EventArgs e)
{
    OpenFileDialog dlg = new OpenFileDialog();
    dlg.Filter = "txt files (*.txt)|*.txt|All files (*.*)|*.*";
    dlg.Multiselect = true; //Câu lệnh cho phép thêm nhiều
    if (dlg.ShowDialog() == DialogResult.OK)
    {
        foreach (string file in dlg.FileNames) //Kiểm tra người dùng đã chọn
        {
            listKetQua.Items.Add(file); //Add từng đường dẫn vào
        }
    }
}

private void btnThuMuc_Click(object sender, EventArgs e)
{
    FolderBrowserDialog f = new FolderBrowserDialog();
    f.ShowDialog();
    listKetQua.Items.Add(f.SelectedPath); //Add đường dẫn
}

private void btnNen_Click(object sender, EventArgs e)
{
    ColorDialog f = new ColorDialog();
    f.ShowDialog();
    listKetQua.BackColor = f.Color; //Đổi màu nền listBox
}

private void btnChu_Click(object sender, EventArgs e)
{
    ColorDialog f = new ColorDialog();
    f.AllowFullOpen = true; //Có allow thì câu lệnh ở dưới mới thực hiện được
```

```
    f.FullOpen = true; //Mở to cái bảng ra
    f.ShowDialog();
    listKetQua.ForeColor = f.Color; //Đổi màu chữ listBox
}

private void btnFont_Click(object sender, EventArgs e)
{
    FontDialog f = new FontDialog();
    f.ShowDialog();
    listKetQua.Font = f.Font;
}

private void btnReset_Click(object sender, EventArgs e)
{
    listKetQua.Items.Clear(); //Xóa bên listBox
    listKetQua.BackColor = Color.Bisque; //Trả về màu nền cũ của listBoxt
    listKetQua.ForeColor = Color.Black; //Trả về màu chữ đen
}
private void btnThoat_Click(object sender, EventArgs e)
{
    if (!aClose)
    {
        timerOut.Enabled = true;
    }
}

private void Form1_Load(object sender, EventArgs e)
{
    //Enabled là một giá trị cho biết dạng thức điều kiện là bật hay tắt.
    //True sẽ bật dạng thức điều kiện. False sẽ tắt dạng thức điều kiện. Mặc định là True.
    timerLoad.Enabled = true;
    aClose = false;
}

private void timerLoad_Tick(object sender, EventArgs e)
{
    if (!aOpen) //Kết quả là false
    {
        this.Opacity += 0.05;
        if (this.Opacity >= 1)
        {
            aOpen = true;
        }
    }
}

private void timerOut_Tick(object sender, EventArgs e)
{
    this.Opacity -= 0.05;
    if (this.Opacity <= 0)
    {
        Application.Exit();
        aClose = true;
    }
}
```

```
    private void Form1_FormClosing(object sender, FormClosingEventArgs e) //Sự kiện khi người dùng nhấn vào X của form
    {
        if (!aClose) //Kết quả là false
        {
            e.Cancel = true;
            timerOut.Enabled = true;
        }
    }
```

Yêu cầu:

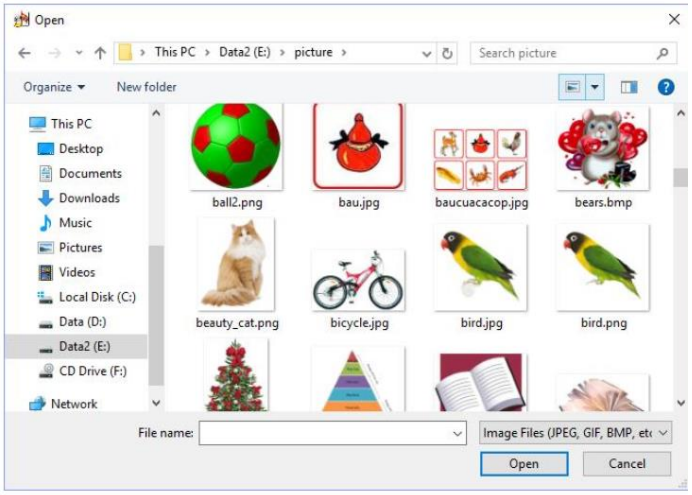
- Menu ban đầu như hình bên dưới:



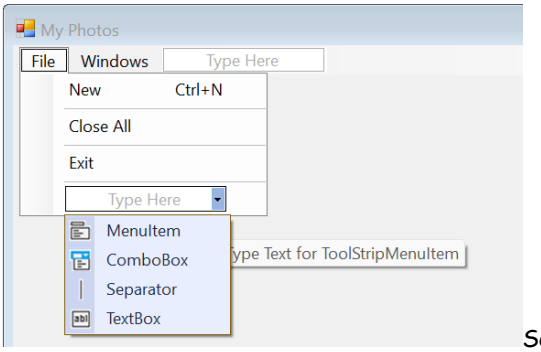
- Click menu File → New: tạo một form con load ảnh một chú mèo

- Sau khi form con được mở, menu sẽ có dạng như sau:

- Open: cho phép mở một file ảnh (chỉ thấy được các file ảnh).



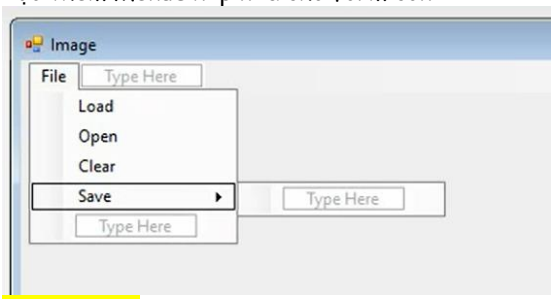
Kéo MenuStrip vào chỉnh



Separator tạo nét rạch

Tạo thêm form con

Tạo thêm MenuStrip nữa cho form con



FORM CON

Tạo pictureBox đặt tên là picImage (chỉnh Dock : canh giữa)

MergeAction form con

MergeAction	MatchOnly

Lần lượt chỉnh các con của menustrip From con : MergerAction về insert (vị trí 1,2,3,4)

Cho memustrip của nó về visible : true

Visible	True

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Drawing.Imaging;

namespace MyPhoTo
{
    public FormChild()
    {
        InitializeComponent();
    }

    private void FormChild_Load(object sender, EventArgs e)
    {
        LoadImage(Application.StartupPath + @"\meo.jpg");
    }
    private void LoadImage(String fileName)
    {
        img = Image.FromFile(fileName);
        pic.Image = img;
    }

    private void MenuLoad_Click(object sender, EventArgs e)
    {
```

Image img;

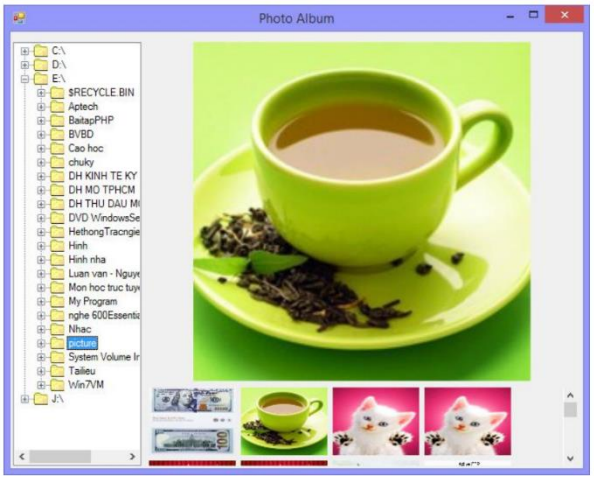
```
            LoadImage(Application.StartupPath + @"\meo.jpg");
        }

        private void MenuOpen_Click(object sender, EventArgs e)
        {
            OpenFileDialog dlg = new OpenFileDialog();
            dlg.Filter = "PNG File (*.png)|*.png"
                + "|JPG File (*.jpg)|*.jpg"
                + "|GIF File (*.gif)|*.gif";
            if (dlg.ShowDialog() == DialogResult.OK)
            {
                LoadImage(dlg.FileName);
            }
        }

        private void MenuClear_Click(object sender, EventArgs e)
        {
            pic.Image = null;
        }

        private void MenuSave_Click(object sender, EventArgs e)
        {
            SaveFileDialog dlg = new SaveFileDialog();
            dlg.Filter = "Image file (jpeg, gif, bmp, ...)|*.jpg;*.gif;*.bmp;*.png|"+"jpeg file(*.jpg)|*.jpg|" +"gif
file(*.gif)|*.gif|" + "bitmap file(*.bmp) | *.bmp|" + "png file (*.png)|*.png";
            if (dlg.ShowDialog() == DialogResult.OK)
            {
                if (dlg.FileName.ToLower().EndsWith(".jpg"))
                {
                    img.Save(dlg.FileName, ImageFormat.Jpeg);
                }
                else if (dlg.FileName.ToLower().EndsWith(".bmp"))
                {
                    img.Save(dlg.FileName, ImageFormat.Bmp);
                }
                else if (dlg.FileName.ToLower().EndsWith(".png"))
                {
                    img.Save(dlg.FileName, ImageFormat.Png);
                } else if (dlg.FileName.ToLower().EndsWith(".gif"))
                {
                    img.Save(dlg.FileName, ImageFormat.Gif);
                }
            }
        }

        private void fileToolStripMenuItem_Click(object sender, EventArgs e)
        {

        }
```

FORM CHA

Lần lượt chỉnh các con của menustrip From cha : MergerAction về insert (vị trí 0,5,6)

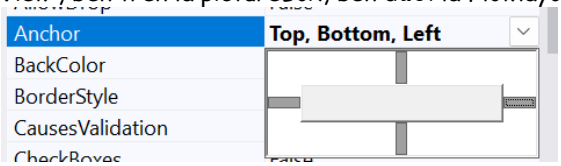
```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.IO;
using static System.Windows.Forms.VisualStyles.VisualStyleElement;

namespace MyPhoTo
```

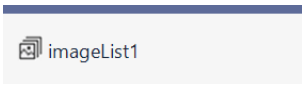
```
private void MenuNew_Click(object sender, EventArgs e)
    {
        IsMdiContainer = true;
        FormChild f = new MyPhoTo.FormChild();
        f.MdiParent = this; //Cho FormChild chạy lòng trong form cha
        f.Show();

    }

    private void cascade_Click(object sender, EventArgs e)
    {
        LayoutMdi(MdiLayout.Cascade);
    }

    private void Horizontal_Click(object sender, EventArgs e)
    {
        LayoutMdi(MdiLayout.TileHorizontal);
    }

    private void Vertical_Click(object sender, EventArgs e)
    {
        LayoutMdi(MdiLayout.TileVertical);
    }

    private void closeAllToolStripMenuItem_Click(object sender, EventArgs e)
    {
        foreach(Form f in MdiChildren)
        {
            f.Close();
        }
    }

    private void exitToolStripMenuItem_Click(object sender, EventArgs e)
    {
        Application.Exit();
    }
```



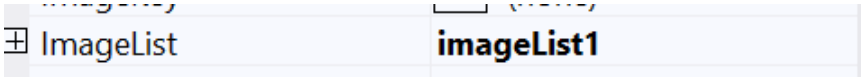
Yêu cầu:
- Khi chạy ứng dụng, danh sách các ổ đĩa trên máy tính sẽ hiển thị trên TreeView.
- Click lên một thư mục có chứa hình ảnh, các hình ảnh trong thư mục đó sẽ hiển thị trên FlowlayoutPanel.
- Chọn một hình trên panel, hình sẽ hiển thị trên picturebox phía trên.

Bên trái là TreeView , bên trên là pictureBox, bên dưới là FlowlayoutPanel

TreeView chỉnh 

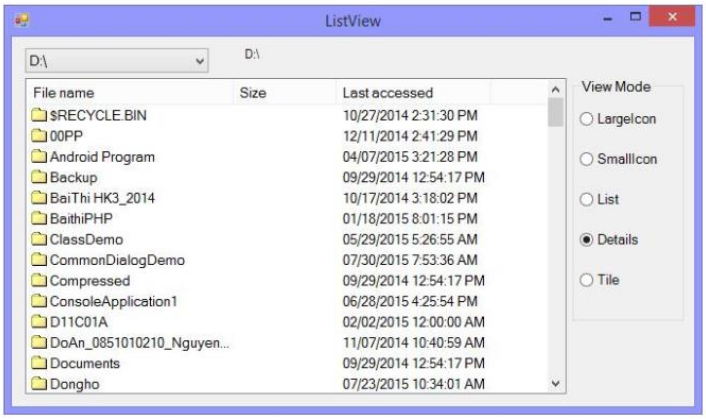
quăng thêm 

add thêm 2 ảnh

TreeView 


```
 private void Form1_Load(object sender, EventArgs e)
      {
         InitTree();
      }

      private void InitTree()
      {
         string[] drive = Directory.GetLogicalDrives();
         TreeNode node = null;
         foreach (string drv in drive)
         {
            node = new TreeNode(drv);
            Tv1.Nodes.Add(node);
            node.Nodes.Add("Temp");
         }
      }

      private void Tv1_BeforeExpand(object sender, TreeViewCancelEventArgs e)
      {
         TreeNode node = e.Node;
         node.Nodes.Clear();
         node.ImageIndex = 0;
         try
         {
            foreach (string dir in Directory.GetDirectories(node.FullPath))
            {
               TreeNode n = node.Nodes.Add(Path.GetFileName(dir));
               n.Nodes.Add("Temp");
            }
```

```
    }
    catch { }
}

private void Tv1_BeforeCollapse(object sender, TreeViewCancelEventArgs e)
{
    e.Node.ImageIndex = 1;
}

private void Tv1_AfterSelect(object sender, TreeViewEventArgs e)
{
    try
    {
        pic1.Image = null;

        string[] arrFile = Directory.GetFiles(e.Node.FullPath);
        layOut1.Controls.Clear();
        foreach(string file in arrFile)
        {
            if(file.ToLower().EndsWith(".png") ||
               file.ToLower().EndsWith(".gif") ||
               file.ToLower().EndsWith(".jpg") ||
               file.ToLower().EndsWith(".bmp") ||
               file.ToLower().EndsWith(".jpeg"))
            {
                PictureBox pic = new PictureBox();

                pic.SizeMode = PictureBoxSizeMode.StretchImage;
                pic.Image = Image.FromFile(file);
                pic.Height = layOut1.Height - 10;
                pic.Width = pic.Height * 5 / 4;
                pic.Cursor = Cursors.Hand;
                layOut1.Controls.Add(pic);

                pic.Click += new EventHandler(pictureBox_Click);
            }
        }
    }
    catch(FormatException)
    {
        MessageBox.Show("Error!");
    }
}

private void pictureBox_Click(object sender, EventArgs e)
{
    PictureBox pic = (PictureBox)sender;
    pic1.Image = pic.Image;
}
```

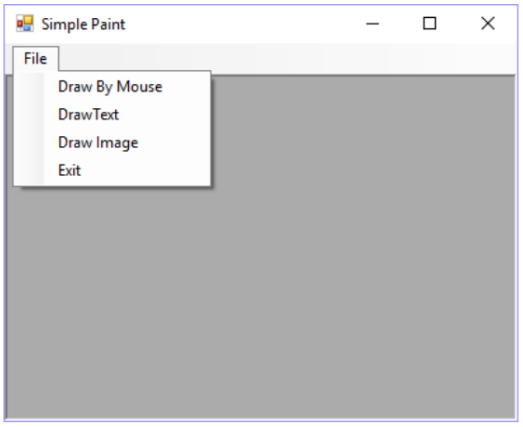



```
string luuPath = null;
      public Form1()
      {
         InitializeComponent();


      }

      private void comboBoxFolder_SelectedIndexChanged(object sender, EventArgs e)
      {
         listViewFolder.Items.Clear();
         try
         {
            if (comboBoxFolder.SelectedItem != null)
            {
               lbFolder.Text = comboBoxFolder.SelectedItem.ToString();
               luuPath = lbFolder.Text;
               foreach (string directory in Directory.GetDirectories(lbFolder.Text))
               {
                  ListViewItem item = new ListViewItem(Path.GetFileName(directory));
                  item.SubItems.Add("");
                  item.SubItems.Add(Directory.GetLastAccessTime(directory).ToString());
                  if (radioButton1.Checked)
                     item.ImageIndex = 2;
                  else if (radioButton3.Checked)
                     item.ImageIndex = 0;
                  listViewFolder.Items.Add(item);
               }
               foreach (string file in Directory.GetFiles(lbFolder.Text))
               {
                  ListViewItem item = new ListViewItem(Path.GetFileName(file));
                  item.SubItems.Add("");
                  item.SubItems.Add(File.GetLastAccessTime(file).ToString());
                  if (radioButton1.Checked)
                     item.ImageIndex = 2;
                  else if (radioButton3.Checked)
                     item.ImageIndex = 0;
                  listViewFolder.Items.Add(item);
               }
            }
            else
            {
```

```
            MessageBox.Show("Please select a folder.");
        }
    }
    catch (Exception ex)
    {
        MessageBox.Show("An error occurred: " + ex.Message);
    }
}

private void Form1_Load(object sender, EventArgs e)
{

    string[] drives = Directory.GetLogicalDrives();
    foreach (string drive in drives)
    {
        comboBoxFolder.Items.Add(drive);
    }
    comboBoxFolder.SelectedIndex = 0;
}


private void radioButton1_CheckedChanged(object sender, EventArgs e)
{
    comboBoxFolder_SelectedIndexChanged(sender, e);
    listViewFolder_DoubleClick(sender, e);
}

private void listViewFolder_DoubleClick(object sender, EventArgs e)
{
    luuPath += @"\" + listViewFolder.SelectedItems[0].Text;
    try
    {
        if (listViewFolder.SelectedItems[0].Text.ToLower().EndsWith(".pdf") ||
            listViewFolder.SelectedItems[0].Text.ToLower().EndsWith(".txt") ||
            listViewFolder.SelectedItems[0].Text.ToLower().EndsWith(".docs") ||
            listViewFolder.SelectedItems[0].Text.ToLower().EndsWith(".doc") ||
            listViewFolder.SelectedItems[0].Text.ToLower().EndsWith(".empl"))
        {
            using (StreamReader read = new StreamReader(luuPath))
            {
                Form2 frm = new Form2();
                frm.data(read.ReadToEnd());
                frm.ShowDialog();
            }
            //sau khi chon xong phai xoa path de chon lai;
            luuPath = luuPath.Substring(0, luuPath.Length - listViewFolder.SelectedItems[0].Text.Length);
        }
        else
        {
            string[] path = Directory.GetDirectories(luuPath);
            listViewFolder.Items.Clear();
            foreach (string drive in path)
            {
                ListViewItem temp = new ListViewItem(Path.GetFileName(drive));
                temp.SubItems.Add("");
```

```
                temp.SubItems.Add(Directory.GetLastAccessTime(drive).ToString());
                if (radioButton1.Checked == true)
                    temp.ImageIndex = 2;
                else if (radioButton3.Checked == true)
                    temp.ImageIndex = 0;
                listViewFolder.Items.Add(temp);
            }
            foreach (string drive in Directory.GetFiles(luuPath))
            {
                try
                {
                    FileInfo f = new FileInfo(luuPath + @"\" + Path.GetFileName(drive));
                    ListViewItem temp = new ListViewItem(Path.GetFileName(drive));
                    temp.SubItems.Add(((double)(f.Length) / 1000).ToString());
                    temp.SubItems.Add(Directory.GetLastAccessTime(drive).ToString());
                    temp.ImageIndex = 1;
                    listViewFolder.Items.Add(temp);
                }
                catch (IOException ex)
                {
                    // xử lý ngoại lệ khi không thể lấy thông tin về file
                    Console.WriteLine("IOException occurred: " + ex.Message);
                }
            }
        }
    }
    catch (FileNotFoundException ex)
    {
        // xử lý ngoại lệ khi không tìm thấy tệp tin
        Console.WriteLine("FileNotFoundException occurred: " + ex.Message);
    }
    catch (IOException ex)
    {
        // xử lý ngoại lệ khi có lỗi trong quá trình đọc tệp tin
        Console.WriteLine("IOException occurred: " + ex.Message);
    }
}
```

- Form chính:



- Menu Draw Text: mở form con vẽ các chuỗi như sau.

- Menu Draw By Mouse: vẽ tự do bằng chuột
  - Giữ chuột trái rê vẽ.
  - Nhấn phím R,G,B để chuyển màu.
  - Nhấn phím ↑: tăng độ dày nét vẽ.
  - Nhấn phím ↓: giảm độ dày nét vẽ.

FORM CHÍNH

Tạo menuStrip

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
private void menuExit_Click(object sender, EventArgs e)
    {
        Application.Exit();
    }

    private void btDrwTxt_Click(object sender, EventArgs e)
    {
        FrmDrwTxt f = new BTH7.FrmDrwTxt();
        f.MdiParent = this;
        f.Show();
    }

    private void btDrwImg_Click(object sender, EventArgs e)
    {
        FrmDrwImg f = new BTH7.FrmDrwImg();
        f.MdiParent = this;
        f.Show();

    }

    private void btDrwByMouse_Click(object sender, EventArgs e)
    {
        FrmDrwMouse f = new BTH7.FrmDrwMouse();
        f.MdiParent = this;
        f.Show();
    }
```

FORM DRAW BY MOUSE

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Drawing.Drawing2D;
Color color;
    int pentWidt;
    Point pOld;
    Bitmap bmp;
    public FrmDrwMouse()
    {
        InitializeComponent();
    }

    private void FrmDrwMouse_Load(object sender, EventArgs e)
    {
```

```
            color = Color.Red;
            pentWidt = 1;
            bmp = new Bitmap(Screen.PrimaryScreen.Bounds.Width, Screen.PrimaryScreen.Bounds.Height);

        }

        private void FrmDrwMouse_MouseDown(object sender, MouseEventArgs e)
        {
            pOld = e.Location;

        }

        private void FrmDrwMouse_MouseMove(object sender, MouseEventArgs e)
        {
             if(e.Button == MouseButtons.Left)
            {
                Pen pen = new Pen(color, pentWidt);
                pen.StartCap = LineCap.Round;
                pen.EndCap = LineCap.Round;
                Graphics g = Graphics.FromImage(bmp);
                g.DrawLine(pen, pOld, e.Location);
                pOld = e.Location;
                Invalidate();
            }
        }

        private void FrmDrwMouse_Paint(object sender, PaintEventArgs e)
        {
            e.Graphics.DrawImage(bmp, 0, 0);
        }

        protected override bool ProcessDialogKey(Keys keyData)
        {
            switch (keyData)
            {
                case Keys.R: color = Color.Red; break;
                case Keys.G: color = Color.Green; break;
                case Keys.B: color = Color.Blue; break;
                case Keys.Up: pentWidt++; break;
                case Keys.Down: pentWidt--; break;
            }
            return true;
        }
    }
```

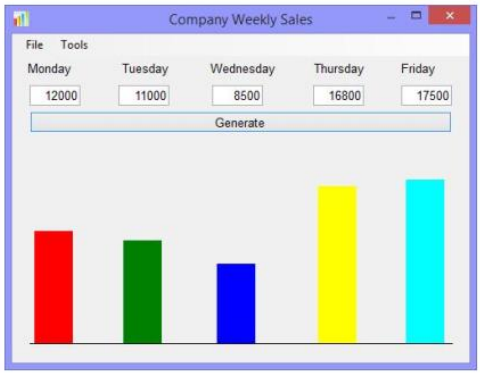
FORM DRAW TEXT

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Drawing.Drawing2D;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
private void FrmDrwTxt_Paint(object sender, PaintEventArgs e)
    {
        Font f = new Font("Arial", 36, FontStyle.Bold);
        StringFormat format = new StringFormat();
        format.Alignment = StringAlignment.Far;
        e.Graphics.DrawString("Hello", f, Brushes.Green, ClientRectangle);

        TextureBrush tbr = new TextureBrush(Image.FromFile(Application.StartupPath + @"\hoa.jpg"));
        format.Alignment = StringAlignment.Near;
        format.LineAlignment = StringAlignment.Far;
        e.Graphics.DrawString("Hello", f,tbr, ClientRectangle, format);

        HatchBrush hbr = new HatchBrush(HatchStyle.DarkHorizontal, Color.Red, Color.Green);
        format.FormatFlags = StringFormatFlags.DirectionVertical;
        //format.LineAlignment = StringAlignment.Near;
        e.Graphics.DrawString("HELLO", f, hbr, ClientRectangle, format);

        LinearGradientBrush lbr = new LinearGradientBrush(new Rectangle(50, 50, 10, 10),
            Color.Blue, Color.White, 45);
        format.Alignment = StringAlignment.Far;
        format.LineAlignment = StringAlignment.Far;
        e.Graphics.DrawString("HELLO", f, lbr, ClientRectangle, format);
    }

    private void FrmDrwTxt_SizeChanged(object sender, EventArgs e)
    {
        Invalidate();
    }
```

FORM DRAW IMAGE

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Drawing.Drawing2D;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
private void FrmDrwImg_Paint(object sender, PaintEventArgs e)
    {
        Rectangle rec1 = new Rectangle(0, 0, ClientRectangle.Width / 2, ClientRectangle.Height / 2);
        Rectangle rec2 = new Rectangle(0, ClientRectangle.Height / 2, ClientRectangle.Width / 2,
ClientRectangle.Height / 2);
```

```
        Rectangle rec3 = new Rectangle(ClientRectangle.Width / 2,0, ClientRectangle.Width / 2,
ClientRectangle.Height / 2);
        DrwImg(rec1, e.Graphics);
        DrwTxt(rec2, e.Graphics);
        DrwPolygon(rec3, e.Graphics);
    }

    private void DrwImg(Rectangle rec , Graphics g)
    {
        Image img = Image.FromFile(Application.StartupPath + @"XucXac\back.png");
        g.DrawImage(img, rec);

        Font f = new Font("Arial", 24, FontStyle.Bold);
        SolidBrush br = new SolidBrush(Color.FromArgb(50, 255, 255, 0));
        StringFormat fm = new StringFormat();
        fm.LineAlignment = StringAlignment.Far;
        g.DrawString("Anita", f, br, rec, fm);
    }
    private void DrwTxt(Rectangle rec, Graphics g)
    {
        LinearGradientBrush br = new LinearGradientBrush(rec, Color.Black, Color.White,45);
        g.FillRectangle(br, rec);
        LinearGradientBrush lbr = new LinearGradientBrush(new Rectangle(0, 0, 40, 40),
            Color.Red,Color.Yellow,LinearGradientMode.BackwardDiagonal);
        StringFormat fm = new StringFormat();
        fm.Alignment = StringAlignment.Center;
        fm.LineAlignment = StringAlignment.Center;
        Font f = new Font("Arial", 48, FontStyle.Bold | FontStyle.Italic);
        g.DrawString("HELLO", f, lbr, rec, fm);

    }
    private void DrwPolygon(Rectangle rec, Graphics g)
    {
        Point[] arrP = { new Point(rec.Left,rec.Height/2) ,
                         new Point(rec.Left + rec.Width/2,0),
                         new Point(rec.Left + rec.Width, rec.Width/4),
                         new Point(rec.Left + rec.Width/2, rec.Width)};
        GraphicsPath path = new GraphicsPath();
        path.AddPolygon(arrP);
        PathGradientBrush br = new PathGradientBrush(path);
        br.CenterColor = Color.White;
        Color[] arrC = { Color.Red, Color.Yellow, Color.Cyan };
        br.SurroundColors = arrC;
        g.FillPolygon(br,arrP);
    }

    private void FrmDrwImg_SizeChanged(object sender, EventArgs e)
    {
        Invalidate();
    }
```



Yêu cầu:

- Nhập các số liệu trong các textbox, click button Generate để vẽ các biểu đồ tương ứng như hình trên.

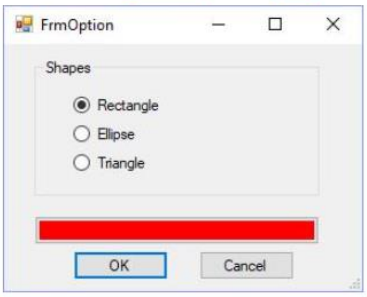
```
Graphics g;//de ve
    Bitmap bmp;// duoc ve
    private void BT2_Load(object sender, EventArgs e)
    {
       t1.Text = "12000";
       t2.Text = "11000";
       t3.Text = "8500";
       t4.Text = "16800";
       t5.Text = "17500";

       bmp = new Bitmap(Screen.PrimaryScreen.Bounds.Width, Screen.PrimaryScreen.Bounds.Height);
       g = Graphics.FromImage(bmp);
    }

    private int GetMaxValue(int[] arrValue)
    {
       int max = 0;
       foreach (int value in arrValue)
       {
          if (value > max) max = value;
       }
       return max;
    }

    private void BT2_Paint(object sender, PaintEventArgs e)
    {
       e.Graphics.DrawImage(bmp, 0, 0);
    }

    private void btGen_Click(object sender, EventArgs e)
    {
       try
       {

          int monday = Convert.ToInt32(t1.Text) / 100;
          int tuesday = Convert.ToInt32(t2.Text) / 100;
          int wednesday = Convert.ToInt32(t3.Text) / 100;
          int thurday = Convert.ToInt32(t4.Text) / 100;
          int friday = Convert.ToInt32(t5.Text) / 100;
          int[] arrValue = { monday, tuesday, wednesday, thurday, friday };
```

```
            int maxValue = GetMaxValue(arrValue);
            if (maxValue * 100 > 50000)
            {
               MessageBox.Show("Vui lòng nhập số nhỏ hơn 50.000", "Thông báo", MessageBoxButtons.OK,
MessageBoxIcon.Error);
               t1.Text = "";
               t2.Text = "";
               t3.Text = "";
               t4.Text = "";
               t5.Text = "";
               t1.Focus();
               return;
            }
            //tao chieu cao cua form dua theo chi so cao nhat
            this.Height = btGen.Bottom + maxValue + 100;
            int bottom = this.Height - 50;

            //Khoi phuc bitmap
            bmp.Dispose();
            bmp = new Bitmap(this.Width, this.Height);


            //khoi phuc graph
            g.Dispose();
            g = Graphics.FromImage(bmp);
            g.Clear(BackColor);

            g.FillRectangle(new SolidBrush(Color.Red),
               t1.Left + 15, bottom - monday, 40, monday);
            g.FillRectangle(new SolidBrush(Color.Green),
               t2.Left + 15, bottom - tuesday, 40, tuesday);
            g.FillRectangle(new SolidBrush(Color.Blue),
               t3.Left + 15, bottom - wednesday, 40, wednesday);
            g.FillRectangle(new SolidBrush(Color.Cyan),
               t4.Left + 15, bottom - thurday, 40, thurday);
            g.FillRectangle(new SolidBrush(Color.Yellow),
               t5.Left + 15, bottom - friday, 40, friday);
            g.DrawLine(new Pen(Color.Black), t1.Left, bottom, t5.Right, bottom);
            Invalidate();

         }
         catch (FormatException)
         {
            MessageBox.Show("Vui lòng nhập đúng định dạng số", "Thông báo", MessageBoxButtons.OK,
MessageBoxIcon.Error);
            t1.Text = "";
            t2.Text = "";
            t3.Text = "";
            t4.Text = "";
            t5.Text = "";
            t1.Focus();
         }
      }
```

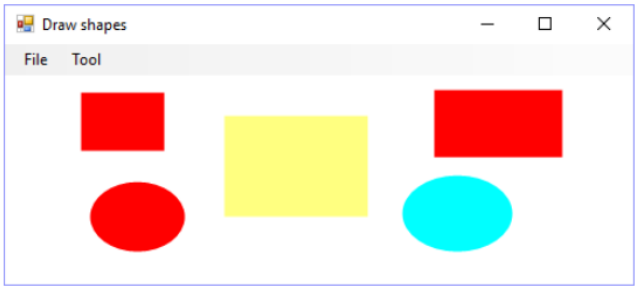
Yêu cầu:

- Giao diện ban đầu: có thể dùng chuột vẽ các hình chữ nhật màu đỏ
- Menu File → Clear All: xóa hết các hình
- Menu File → Exit: đóng ứng dụng

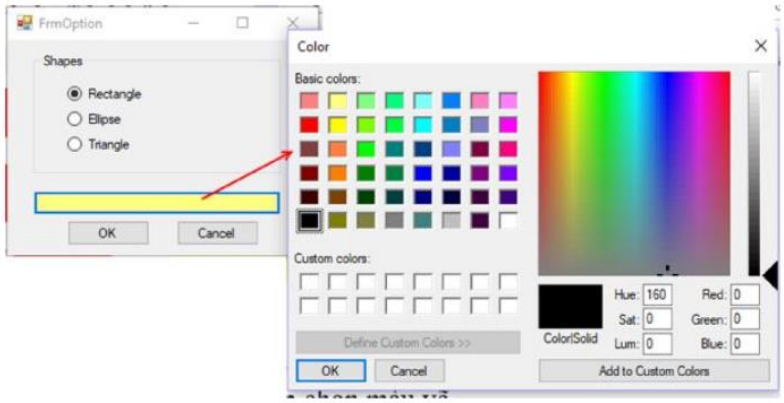
- Menu Tool →Option: mở hộp thoại cho phép chọn loại hình vẽ và màu tô:



- Rectangle: vẽ hình chữ nhật
- Ellipse: vẽ ellipse
- Triangle: vẽ tam giác

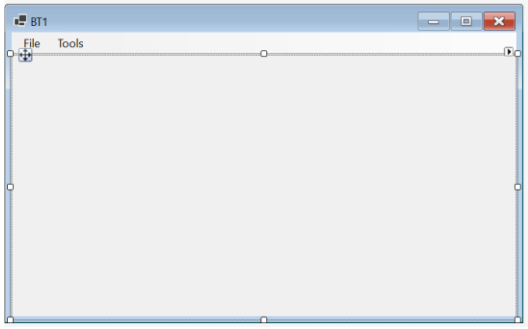


- Chọn màu: mở hộp thoại cho phép chọn màu vẽ



FORM CHÍNH

Tạo menuStrip và panel



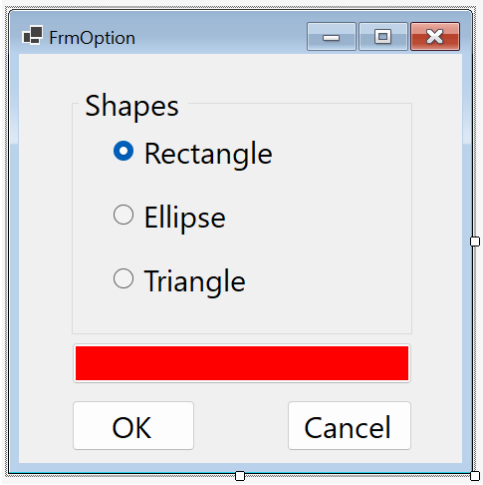

```
private List<Rectangle> rectangles;
        private List<Rectangle> ellipse;
        private List<Point[]> triangle;
        private Point startPoint;
        private Point endPoint;
        private bool drawing;
        public BT1()
        {
            InitializeComponent();
            rectangles = new List<Rectangle>();
            ellipse = new List<Rectangle>();
            triangle = new List<Point[]>();
            startPoint = Point.Empty;
            endPoint = Point.Empty;
            drawing = false;
        }


        private void BT1_Paint(object sender, PaintEventArgs e)
        {

        }

        private void panel1_MouseUp(object sender, MouseEventArgs e)
        {
            if (FrmOption.i == 1)
                RectUp();
            else if (FrmOption.i == 2)
                EllipseUp();
            else if (FrmOption.i == 3)
```

```
        TriangleUp(e);
}
private void RectUp()
{
    // Khi chuột được thả, dừng vẽ hình và lưu hình chữ nhật vào danh sách
    drawing = false;
    Rectangle rect = new Rectangle(
        Math.Min(startPoint.X, endPoint.X),
        Math.Min(startPoint.Y, endPoint.Y),
        Math.Abs(startPoint.X - endPoint.X),
        Math.Abs(startPoint.Y - endPoint.Y));
    rectangles.Add(rect);
}

private void EllipseUp()
{
    // Khi chuột được thả, dừng vẽ hình và lưu hình ellipse vào danh sách
    drawing = false;
    Rectangle rect = new Rectangle(
        Math.Min(startPoint.X, endPoint.X),
        Math.Min(startPoint.Y, endPoint.Y),
        Math.Abs(startPoint.X - endPoint.X),
        Math.Abs(startPoint.Y - endPoint.Y));
    ellipse.Add(rect);
}

private void TriangleUp(MouseEventArgs e)
{
    drawing = false;
    endPoint = e.Location;
    Point[] points = { startPoint, new Point(endPoint.X, startPoint.Y), endPoint };
    triangle.Add(points);
    panel1.Invalidate();
}
private void panel1_MouseMove(object sender, MouseEventArgs e)
{
    // Nếu đang vẽ hình, lưu tọa độ điểm kết thúc và vẽ hình lên màn hình
    if (drawing)
    {
        endPoint = e.Location;
        panel1.Invalidate();
    }
}

private void panel1_MouseDown(object sender, MouseEventArgs e)
{
    // Khi chuột được nhấn, lưu tọa độ điểm bắt đầu
    startPoint = e.Location;
    drawing = true;
}

private void panel1_Paint(object sender, PaintEventArgs e)
{

        Rectangle(e);
```

```
        Ellipse(e);
        Triangle(e);

}

private void clearAllToolStripMenuItem_Click(object sender, EventArgs e)
{
   rectangles.Clear();
   ellipse.Clear();
   triangle.Clear();
   panel1.Invalidate();

}

private void optionsToolStripMenuItem1_Click(object sender, EventArgs e)
{
   FrmOption f = new FrmOption();
   f.ShowDialog();

}

private void BT1_Load(object sender, EventArgs e)
{
   FrmOption.cl = Color.Red;
   FrmOption.i = 1;
}

private void Rectangle(PaintEventArgs e)
{
   // Vẽ tất cả các hình chữ nhật đã lưu trữ
   foreach (Rectangle rect in rectangles)
   {
      SolidBrush brush = new SolidBrush(FrmOption.cl);
      e.Graphics.FillRectangle(brush, rect);
   }

   // Nếu đang vẽ hình mới, vẽ hình đó lên màn hình
   if (drawing)
   {
      Rectangle rect = new Rectangle(
         Math.Min(startPoint.X, endPoint.X),
         Math.Min(startPoint.Y, endPoint.Y),
         Math.Abs(startPoint.X - endPoint.X),
         Math.Abs(startPoint.Y - endPoint.Y));
      SolidBrush brush = new SolidBrush(FrmOption.cl);
      e.Graphics.FillRectangle(brush, rect);
   }
}

private void Ellipse(PaintEventArgs e)
{
   // Vẽ tất cả các hình ellipse đã lưu trữ
   foreach (Rectangle rect in ellipse)
   {
      SolidBrush brush = new SolidBrush(FrmOption.cl);
```

```
            e.Graphics.FillEllipse(brush, rect);
        }

        // Nếu đang vẽ hình mới, vẽ hình đó lên màn hình
        if (drawing)
        {
            Rectangle rect = new Rectangle(
                Math.Min(startPoint.X, endPoint.X),
                Math.Min(startPoint.Y, endPoint.Y),
                Math.Abs(startPoint.X - endPoint.X),
                Math.Abs(startPoint.Y - endPoint.Y));
            SolidBrush brush = new SolidBrush(FrmOption.cl);
            e.Graphics.FillEllipse(brush, rect);
        }
    }

    private void Triangle(PaintEventArgs e)
    {
        // Vẽ các hình tam giác
        foreach (Point[] points in triangle)
        {
            Brush brush = new SolidBrush(FrmOption.cl);
            e.Graphics.FillPolygon(brush, points);
        }

        // Vẽ tam giác đang được vẽ
        if (drawing)
        {
            Point[] points = { startPoint, new Point(endPoint.X, startPoint.Y), endPoint };
            Brush brush = new SolidBrush(FrmOption.cl);
            e.Graphics.FillPolygon(brush, points);
        }

    }
```

FORM PHỤ



Thanh màu là button

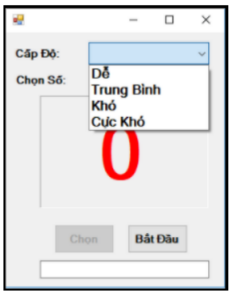
```
public static int i;

    private void btOk_Click(object sender, EventArgs e)
    {
        if (btRect.Checked)
```

```
            i = 1;
        else if (btEllipse.Checked)
            i = 2;
        else if (btTrian.Checked)
            i = 3;
        this.Close();
    }
    public static Color cl;
    private void btColor_Click(object sender, EventArgs e)
    {
        ColorDialog cld = new ColorDialog();
        cl = new Color();
        if (cld.ShowDialog() == DialogResult.OK)
        {
            cl = cld.Color;
            btColor.BackColor = cld.Color;

        }

    }
```

LÝ THUYẾT

1. ComboBox

ComboBox được dùng để hiển thị một danh sách những mỗi lần người dùng chỉ có thể chọn một lựa chọn, có thể nhập mới.



Một số thao tascc với ComboBox:

Add(): Thêm một mục chọn vào cuối danh sách ListBox.
Insert(): Chèn thêm mục chọn vào vị trí **i**.
Count: Trả về số muc chọn hiện đang có.
Item(): Trả về mục chọn ở vị trí thứ **i**.
Remove(): Bỏ mục chọn.
RemoveAt(): Bỏ mục chọn ở vị trí thứ **i**.
Contains(): Trả về True nếu có mục chọn trong danh sách, trả về False nếu không có mục chọn trong danh sách.
Clear: Xóa tất cả các mục chọn.
IndexOf(): Trả về vị trí mục chọn trong danh sách, nếu không tìm thấy sẽ trả về -1.
Một số thuộc tính thương dùng:

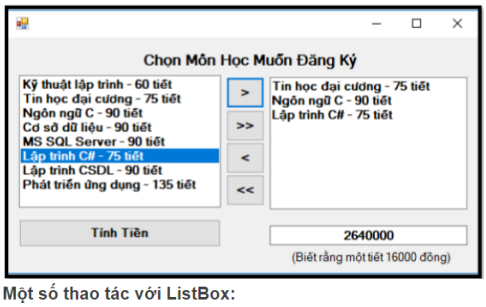
Một số thuộc tính thương dùng:

Thuộc tính	Mô tả
Text	Trả về nội dung dòng dữ liệu đang hiển thị trên ComboBox
DropDownStyle	Quy định định dạng của ComboBox, nhận một tring các giá trị: - Simple: hiển thị theo dạng ListBox + TextBox có thể chọn dữ liệu từ ListBox hoặc nhập mới vào TextBox - DropDownList: Chỉ cho phép chọn dữ liệu trong ComboBox - DropDown: Giá trị mặc định, có thể chọn hoặc nhập mới mục dữ liệu vào ComboBox
Items	Trả về các mục chứa trong ComboBox
DropDownHeight	Thiết lập chiều cao tối đa khi sổ xuống của ComboBox
DropDownWidth	Thiết lập độ rộng của mục chọn trong ComboBox
SelectedIndex	Lấu chỉ số mục được chọn, chỉ số mục đầu tiên là 0
SelectedItem	Trả về mục được chọn
SelectedText	Lấy chuỗi hiển thị của mục chọn trên ComboBox
DataSource	Chọn tập dữ liệu điền vào ComboBox. Tập dữ liệu có thể là mảng, chuỗi, ArrayList,...
DisplayMember	Gán dữ liệu thành viên sẽ hiển thị trên ComboBox
ValueMember	Thuộc tính này chỉ định dữ liệu thành viên sẽ cung cấp giá trị cho ComboBox
SelectedValue	Trả về giá trị của mục chọn (ValueMember) nếu ComboBox có liên kết dữ liệu. Nếu không liên kết dữ liệu hoặc ValueMember không được thiết lập thì giá trị SelectedValue là giá trị chuỗi của thuộc tính SelectedItem

Trong **ComboBox** có một sự kiện là ***SelectedIndexChanged***, sự kiện này xảy ra khi thay đổi mục chọn trong ComboBox.

2. ListBox

ListBox được dùng để hiển thị một danh sách các lựa chọn, người dùng có thể chọn một hoặc nhiều lựa chọn cùng lúc.



Một số thao tác với ListBox:

Add(): Thêm một mục chọn vào cuối danh sách **ListBox**.
Insert(): Chèn thêm mục vào vào vị trí **i**.
Count: Trả về số mục chọn hiện đang có.
Item(): Trả về mục chọn ở vị trí **i**.
Remove(): Bỏ mục chọn.
RemoveAt(): Bỏ mục chọn tại vị trí **i**.
Contains(): Trả về True nếu có mục chọn trong danh sách và trả về False nếu không có mục chọn trong danh sách.
Clear: Xóa tất cả các mục chọn.
IndexOf(): Trả về vị trí mục chọn trong danh sách, nếu không tìm thấy sẽ trả về -1.

Một số thuộc tính của ListBox thường dùng:

Thuộc tính	Mô tả
DataSource	Chọn tập dữ liệu điền vào ListBox. Tập dữ liệu có thể là mảng, chuỗi, ArrayList,...
DisplayMember	Dữ liệu thành viên sẽ được hiển thị trên ListBox
ValueMember	Thuộc tính này chỉ định dữ liệu thành viên sẽ cung cấp giá trị cho ListBox
SelectedValue	Trả về giá trị của mục chọn nếu ListBox có liên kết dữ liệu. Nếu không liên kết với dữ liệu hoặc thuộc tính ValueMember không được thiết lập thì giá trị thuộc tính SelectedValue là giá trị chuỗi của thuộc tính SelectedItem
Items	Các mục chứa trong ListBox
SelectedItem	Trả về mục được chọn
SelectedIndex	Lấy chỉ số mục được chọn, chỉ số mục chọn đầu tiên là 0
SelectionMode	Cho phép chọn một hoặc nhiều dòng dữ liệu trên ListBox, bao gồm: - *One*: Chỉ chọn một giá trị - *MultiSimple*: Cho phép chọn nhiều, chọn bằng cách Click vào mục chọn, bỏ chọn bằng cách Click vào mục đã chọn - *MultiExtended*: Chọn nhiều bằng cách nhấn kết hợp với Shift hoặc Ctrl
SelectedItems	Được sử dụng khi SelectionMode là MultiSimple hoặc MultiExtended. Thộc tính SelectedItems chứa các chỉ số của các dòng dữ liệu được chọn
SelectedItems	Được sử dụng khi SelectionMode là MultiSimple hoặc MultiExtended. Thuộc tính SelectedItems chứa các chỉ số của các dòng dữ liệu được chọn

Trong ListBox có một sự kiện được sử dụng rất nhiều đó chính là ***SelectedIndexChanged***, sự kiện này xảy ra khi thay đổi mục chọn trong ListBox.

Bước 3: Xử lý sự kiện cho nút "**Thoát**".

Đối với sự kiện này thì ta đã xử lý khá nhiều, vì đa số trong các ứng dụng đều có sự kiện thoát giúp người dùng thoát khỏi chương trình.

```
private void btnOut_Click(object sender, EventArgs e)
    {
        DialogResult dg = MessageBox.Show("Bạn có muốn đóng chương trình", "Thông báo",
MessageBoxButtons.YesNo, MessageBoxIcon.Question);
        if(dg == DialogResult.Yes)
        {
            Application.Exit();
        }
    }
```

MDI (Multiple Document Interface) là kiểu giao diện người dùng (GUI – Graphic User Interface) trong đó có nhiều cửa sổ con được cư trú trong cửa sổ cha duy nhất, ngược lại với kiểu giao diện SDI (Single Document Interface) mà tất cả các cửa sổ là độc lập với nhau. Ví dụ Notepad của Windows là dạng giao diện SDI, Visual Studio là dạng giao diện MDI. Hôm nay chúng ta sẽ xây dựng chương trình giống như Notepad của Window nhưng dạng giao diện MDI, chúng ta đặt tên chương trình này là MDINotepad.

IsMdiContainer = true (cho phép 1 form cha chứa nhiều form con)

```
    //Tạo biến chạy cho text box
    1 reference
    private void FormDangNhap_Load(object sender, EventArgs e)
    {
        lbName.Text = "     ~~ LOGIN AS ADMIN ~~     ";
        txtDN.Focus();
    }

    1 reference
    private void timer1_Tick(object sender, EventArgs e)
    {
        lbName.Text = lbName.Text.Substring(1) + lbName.Text.Substring(0, 1);
    }
```

----------CHUYỂN ĐỔI KIỂU DỮ LIỆU VÀ ÉP KIỂU ----------

1.PHƯƠNG THỨC Parse

- Sử dụng khá phổ biến khi chúng ta muốn chuyển đổi một chuỗi sang một kiểu dữ liệu tương ứng.

Boolean.Parse dùng để chuyển về kiểu Boolean

Int32.Parse dùng để chuyển về kiểu Int 32

Double.Parse dùng để chuyển chuỗi về kiểu Double

VD:

int a = Int32.Parse("123"); //a sẽ mang giá trị 123

float b = Float.Parse("20.7"); //b sẽ mang giá trị 20.7

bool c = Boolean.Parse("true"); //c sẽ mang giá trị true

- Nếu như chuỗi chúng ta truyền vào là rỗng, không đúng định dạng hoặc vượt quá giá trị cho phép thì chúng ta sẽ nhận được các Exception tương ứng.

VD:

int a = Int32.Parse(“Hello”); //sai định dạng, FormatException

byte b = Byte.Parse(“10000000000”); //quá giới hạn, OverflowException

bool c = Boolean.Parse(null); //tham số là null, ArgumentNullException

2.PHƯƠNG THỨC TryParse

- Là phương thức được tích hợp sẵn trong các lớp kiểu dữ liệu cơ bản của C#. Tuy nhiên, cú pháp của TryParse có phần khác với Parse. Cụ thể, tham số thứ nhất của TryParse là chuỗi cần chuyển đổi và tham số thứ hai là biến sẽ chứa giá trị đã được chuyển đổi, biến thứ hai này phải được đánh dấu là out (để cho biến là chúng ta sẽ truyền tham chiếu)

VD:

int a;

Int32.TryParse(“123”, out a); //a sẽ mang giá trị 123

bool b;

Boolean.TryParse(“false”, out b); //b sẽ mang giá trị false

- Phương thức TryParse sẽ thực thi nhanh hơn phương thức Parse vì TryParse không ném ra ngoại lệ

3. LỚP Convert

- Là một lớp tiện ích trong C# cung cấp cho chúng ta rất nhiều phương thức tĩnh khác nhau để chuyển đổi từ một kiểu dữ liệu này sang kiểu dữ liệu khác. Tham số mà các phương thức trong Convert nhận không nhất thiết phải là chuỗi mà có thể ở nhiều kiểu dữ liệu khác nhau (int, bool, double…).
VD:
int a = Convert.ToInt32("123"); //chuyển chuỗi 123 sang số nguyên
bool b = Convert.ToBoolean(27); //chuyển số 27 sang kiểu bool
- Các phương thức trong lớp Convert sẽ trả về giá trị mặc định nếu như tham số truyền vào là null. Còn trong các trường hợp sai định dạng hoặc vượt quá giới hạn thì các phương thức đó sẽ ném ra các ngoại lệ tương tự như phương thức Parse.
VD:
bool a = Convert.ToBoolean("hello"); //FormatException
int b = Convert.ToInt32("123456787654"); //OverflowException
double d = Convert.ToDouble(null); //trả về giá trị mặc định

4. Casting (Ép kiểu)

- Ép kiểu là cách chúng ta có thể sử dụng khi muốn chuyển đổi giữa các kiểu dữ liệu có tính chất tương tự nhau (thường là số).
VD:
int a = 100;
float b = a; //chuyển đổi ngầm định, b = 100
int c = (int)b; //chuyển đổi rõ ràng, c = 100
- Ngoài ra, đối với các giá trị được lưu trong kiểu tổng quát Object (bằng cách boxing) thì chúng ta có thể ép kiểu đưa về kiểu dữ liệu ban đầu
VD:
int a = 100;
object b = a; //boxing, b là kiểu tham chiếu chứa giá trị 100
int c = (int)b; //unboxing, c mang giá trị 100
- Ép kiểu chỉ được sử dụng khi chúng ta biết rõ rằng đối tượng đó chứa kiểu dữ liệu tương ứng với kiểu mà ta cần chuyển tới. Ví dụ như các trường hợp sau sẽ là các lỗi cú pháp trong lập trình:
VD:
string a = "1234";
int b = (int)a; //lỗi, không thể ép kiểu chuỗi sang kiểu số
bool c = true;
double d = (double)c; //lỗi, không thể ép kiểu bool sang kiểu double
---------- Sự khác biệt giữa int, Int16, Int32 và Int64 ----------

Int16 -- (-32,768 to +32,767)

Int32 -- (-2,147,483,648 to +2,147,483,647)

Int64 -- (-9,223,372,036,854,775,808 to +9,223,372,036,854,775,807)

int và Int32 thực sự là đồng nghĩa; int sẽ trông quen thuộc hơn một chút, Int32 làm cho 32 bit rõ ràng hơn đối với những người đọc mã của bạn.
Int16: 2 byte
Int32 và int: 4 byte
Int64 : 8 byte
---------- EXCEPTION ----------
VD: Tự quăng và lỗi format

```
try
{
    if (txtKyTu.Text == "")
        throw new Exception("Phải nhập dữ liệu");
    char kt = char.Parse(txtKyTu.Text);
    txtKQ1.Text = string.Format("{0}", (int)kt);

}
```

```
    catch (FormatException)
        {
            MessageBox.Show("Phải nhập ký tự!");
        }
    catch (Exception ex) //Có này thì cái throw new Exception "Phải nhập dữ liệu" mới diễn ra
        {
            MessageBox.Show(ex.Message);
        }
```

VD: Lỗi vượt quá dữ liệu

```
        catch (DivideByZeroException
```

VD: Lỗi chia cho 0

```
        catch (DivideByZeroException)
```

---------- MESSAGEBOX ----------

- Loại 1: Chỉ có thông tin

VD: MessageBox.Show("Xin chào! Tôi là C#");

- Loại 2: Có thông tin và tiêu đề

VD: MessageBox.Show("Xin chào! Tôi là C#","Thông báo"); (Tiêu đề là thông báo)

- Loại 3: Có thông tin, tiêu đề, nút bấm : MessageBoxButtons.<loại nút>

+ AbortRetryIgnore, OK, OKCancel, RetryCancel, YesNo

VD: MessageBox.Show("Xin chào! Tôi là C#", "Thông báo", MessageBoxButtons.AbortRetryIgnore);

- Loại 4: Có thông tin, tiêu đề, nút bấm và icon

+ Để thêm vào icon ta thêm tham số kiểu enum là MessageBoxIcon.<loại icon>, có nhiều loại nhưng phổ biến là Warning (tam giác vàng có dấu chấm than), Error (hình tròn đỏ có chữ X), Information (hình tròn xanh lam có chữ i), Question (hình tròn lam có dấu chấm hỏi).

VD: MessageBox.Show("Xin chào! Tôi là C#", "Thông báo, MessageBoxButtons.OKCancel, MessageBoxIcon.Question);

---------- string.Format() ----------

- String.Format() là phương thức định dạng chuỗi xuất hiện từ phiên bản đầu tiên của C# và vẫn còn tiếp tục được sử dụng và giới thiệu trong các sách dạy lập trình C#.

VD:

```
var name = "Donald Trump";
var town = "Washington DC";
var age = 18;
var message = string.Format("My name is {0}. I'm from {1}. I'm {2} years old", name, town, age);
```

Trong chuỗi trên, {0}, {1}, {2} được gọi là các placeholder. Giá trị của các biến sẽ được thay thế vào đúng vị trí tương ứng với placeholder: name là biến đầu tiên => số 0 => giá trị của nó sẽ điền vào chỗ {0}. Tương tự với các biến còn lại.

VD:

```
kq = string.Format("Phương trình có 2 nghiệm x1 = {0:0.00}, x2 = {1: 0.00}", x1, x2);
//x1 sẽ là ngiệm của {0} và x2 là nghiệm của {1} 0.00 định dạng số nguyên lấy 2 chữ số thập phân
```

---------- CHUỖI KÝ TỰ STRING TRONG C# (SHARP) ----------

- Chuỗi là một tập hợp các ký tự sắp xếp có vị trí, nó chỉnh là một mảng các ký tự, kiểu dữ liệu chuỗi đó là string, lớp biểu diễn các chuỗi là System.String

VD:

```
string sExample = "Xin chào";       // Khai báo và khởi tạo chuỗi
sExample += " các bạn";             // Nối chuỗi +=, trả về "Xin chào các bạn"
sExample = sExample + "!";          // Nối chuỗi +, trả về "Xin chào các bạn!"
```

- Chuỗi như là mảng mà phần tử mảng là các ký tự, nên có thể truy cập phần tử mảng bằng indexer để đọc ký tự:

VD: char c = sExample[1];   // c= 'i'

---------- Viết chuỗi nguyên bản với ký hiệu @ trong C# ----------

- Khi viết chuỗi trong cặp dấu nháy kép "", thì các ký tự đặt biệt được xử lý với ký hiệu \

VD: string s = "C:\\Abc\\xyz"; // Nếu viết string s = "C:\Abc\xyz"; sẽ lỗi

Nội dung thực tế của chuỗi là C:\Abc\xyz

- Nếu muốn viết chuỗi cố định, nội dung nguyên bản - cho biết sẽ không dùng \ để xử lý ký tự đặc biệt, thì thêm @ vào đầu chuỗi:
VD: string s = @"Ký tự \ được dùng để chèn ký tự đặc biệt như \n, \r";
- Bằng ký hiệu @ chuỗi viết thế nào thì nội dung thực tế sẽ như vậy, ngoại trừ hai ký tự "" chuyển thành một ký tự ""
VD: string s = "Anh ấy nói, ""Đây là C#"""; //~ Anh ấy nói "Đây là C#"
- Ngoài ra bạn có thể viết chuỗi trên nhiều dòng với ký hiệu @
VD: string s = @"Xin chào các bạn
Tôi đang học C#";
---------- Chèn thêm biểu thức vào chuỗi với ký hiệu $ trong C# ----------
- Khi viết chuỗi có ký tự $ phía trước, thì trong chuỗi đó có thể chèn các biểu thức vào chỗ có cặp {}
VD:
```
int a = 10;
int b = 2;
string s = $"Kết quả {a}/{b} là {a/b}";  // "Kết quả 10/2 là 5"
```
- Ngoài ra bạn có thể căn lề, định dạng số, ngày tháng ... tương tự như chuỗi định dạng
VD:
```
Console.WriteLine($"{"VòngLặp",10} {"Chẵn/Lẻ", -5}");
for (int i = 8; i < 15; i++)
{
   string chanle = (i%2 == 0) ? "Chẵn" : "Lẻ";
   Console.WriteLine($"{i,10} {chanle, -5}");
}
```

Kết quả:
```
  VòngLặp Chẵn/Lẻ
         8 Chẵn
         9 Lẻ
        10 Chẵn
        11 Lẻ
        12 Chẵn
        13 Lẻ
        14 Chẵn
```
---------- Một số phương thức làm việc với chuỗi C# ----------
```
string stringA =  "Xin chào,";
string stringB = "các bạn!";
```
-Concat: phương thức tĩnh, nối các chuỗi liệt kê ở tham số lại với nhau
VD: string s = String.Concat(stringA, stringB); // s = "Xin chào,các bạn!"
-Format:
VD: string s = String.Format("Chào {0}, {0} ơi, hôm nay ngày {1} rồi!", "Nam", DateTime.Now.Day); // s = "Chào Nam, Nam hôm nay ngày 20 rồi!"
-IndexOf: Tìm vị trí (đầu tiên) của ký tự hoặc chuỗi ký tự trong chuỗi
-LastIndexOf: Tìm vị trí (cuối) của ký tự hoặc chuỗi ký tự trong chuỗi
-Insert: Tạo chuỗi = chèn chuỗi này vào trong chuỗi khác, vị trí chèn cần chỉ ra
VD: var s = stringA.Insert(8, " tất cả"); // "Xin chào tất cả,"
-PadLeft: Tạo chuỗi mới từ chuỗi cũ, độ dài chuỗi mới chỉ ra - nếu độ dài chuỗi mới lớn hơn chuỗi cũ thì các ký tự phía đầu được chèn khoảng trắng hoặc ký tự chỉ định.
VD:
```
string s1 = "Abc";
string s2 = s1.PadLeft(6);       //  "   Abc"
string s3 = s1.PadLeft(6, '*');   //  "***Abc"
```
-PadRight: Tương tự PadLeft nhưng chèn khoảng trắng bên phải
-Replace: Tìm và thay thế trong chuỗi
VD: var s = stringA.Replace("chào", "CHÀO");   // "Xin CHÀO,"
-Split: Trả về mảng các chuỗi con được chia từ chuỗi gốc bởi ký tự chia chỉ định, chuỗi chia chỉ định

VD: var s = "Nguyễn Văn A".Split(' '); // s tương đương mảng {"Nguyễn","Văn", "A"}

-ToLower: Sinh chuỗi mới bằng cách chuyển các ký tự thành chữ thường

-ToUpper: Sinh chuỗi mới bằng cách chuyển các ký tự thành chữ in

-Trim: Sinh chuỗi mới bằng cách loại bỏ khoảng trắng (hoặc chỉ định) ở đầu và cuối

-Substring :Lấy ra chuỗi con từ chuỗi chính - chuỗi con lấy từ vị trí chỉ ra đến cuối hoặc theo độ dài

VD:

```
string s = stringA.Substring(4); // s = "chào,"
string x = stringA.Substring(0, 3); // s = "Xin" (dài 3)
```

---------- Sử dụng StringBuilder ----------

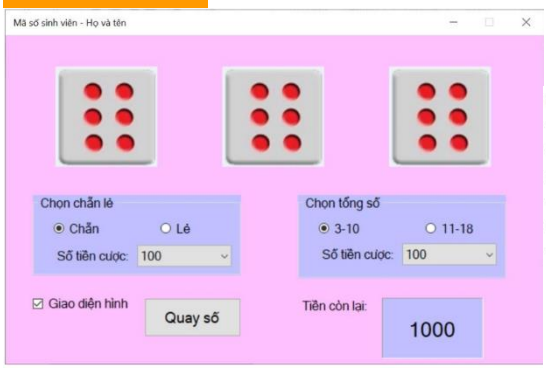
-Khi sử dụng biến kiểu string để thực hiện các thao tác nhằm mục đích cuối cùng thu được chuỗi theo yêu cầu, trong quá trình đó bạn có thể sử dụng nhiều biến kiểu string, để phục vụ các phép toán như nối chuỗi, tìm kiếm, thay thế ... Mỗi khi khởi tạo một biến kiểu string, bạn đã cấp phát một lượng bộ nhớ để lưu trữ chuỗi - thường thì bộ nhớ này nhiều hơn những gì bạn cần.

-Để thi hành tối ưu hơn về tốc độ, về sử dụng bộ nhớ có thể dùng tới đối tượng StringBuilder ở namespace System.Text (thêm vào đầu file using System.Text;)

VD:

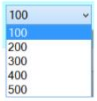
```
StringBuilder stringBuilder = new StringBuilder();
stringBuilder.Append("Xin chào các bạn - xuanthulab.net");
stringBuilder.Replace("Xin chào", "XIN CHÀO");
Console.WriteLine(stringBuilder); // Out: XIN CHÀO các bạn - xuanthulab.net
```

ĐỀ THI GK



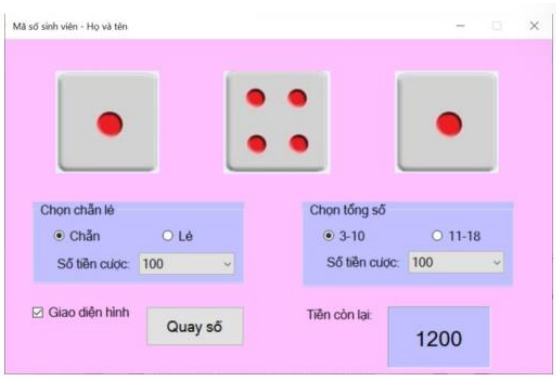
Giao diện ban đầu (3 đ):

- Form hiển thị giữa màn hình, không cho phép thay đổi kích thước, nút phóng to bị vô hiệu hóa
- Tiêu đề Form hiển thị Mã số - Họ tên sinh viên
- CheckBox Giao diện hình được chọn, hiển thị hình ba con xucxac số 6
- Các radiobutton **Chẵn** và **3-10** được chọn mặc định
- Số tiền ban đầu là 1000
- Hai combobox **Số tiền cược** chứa các giá trị từ 100-500, chọn mặc định 100

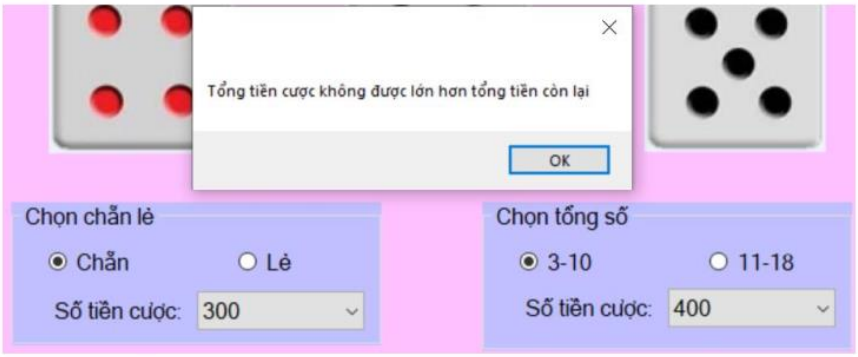


Xử lý ứng dụng:

- Check/Bỏ Check trên checkbox Giao diện hình, các số thay đổi cách hiển thị dạng hình/số (2 điểm)

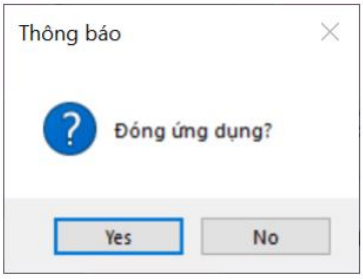


- Xử lý nút **Quay số**: Người chơi chọn **Chẵn/Lẻ**, số tiền cược bên khung **Chọn chẵn lẻ** và chọn **3-10 /11-18**, số tiền cược bên khung **Chọn tổng số**, sau đó nhấn nút **Quay số**:
  - Nếu tổng số tiền đặt cược < tổng số tiền còn lại, hiển thị thông báo như hình bên dưới và không xử lý (1 điểm)



- Nếu số tiền đặt cược phù hợp, sau khi quay số và tính toán kết quả, tiền còn lại sẽ được cập nhật đúng trên giao diện (3,5 điểm)

Đóng ứng dụng: hiển thị hộp thoại cảnh báo (0,5 đ)



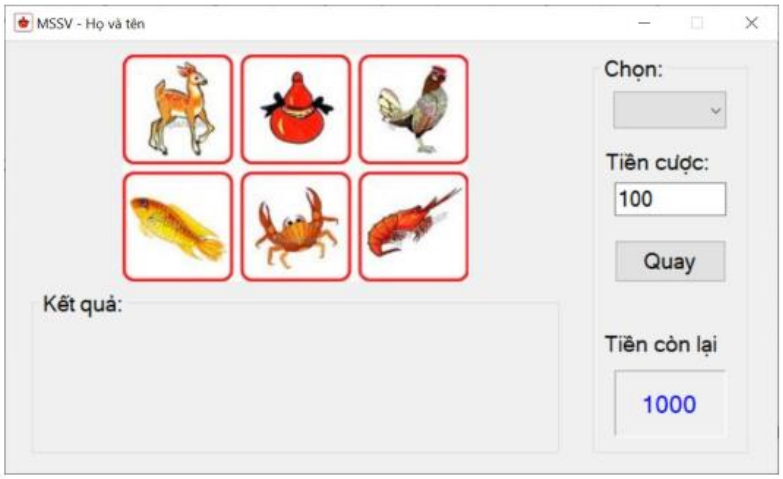

```
string pathImg;
    Random rand = new Random();
    private int soTienConLai;
    public Form1()
    {
        InitializeComponent();
    }

    private void Form1_Load(object sender, EventArgs e)
    {
        pathImg = Application.StartupPath + @"\HinhXucXac\";
        soTienConLai = Convert.ToInt32(lbResult.Text);
        comboBox1.SelectedIndex = 0;
        comboBox2.SelectedIndex = 0;
        label6.Text = label7.Text = label8.Text = "6";
    }
    private void btQuaySo_Click(object sender, EventArgs e)
    {
        int so1 = rand.Next(1, 7);
        int so2 = rand.Next(1, 7);
        int so3 = rand.Next(1, 7);
        label6.Text = so1.ToString();
        label7.Text = so2.ToString();
        label8.Text = so3.ToString();
        pic1.Image = Image.FromFile(pathImg + so1.ToString() + ".gif");
        pic2.Image = Image.FromFile(pathImg + so2.ToString() + ".gif");
        pic3.Image = Image.FromFile(pathImg + so3.ToString() + ".gif");
        int total = so1 + so2 + so3;

        int tienCuoc1 = int.Parse(comboBox1.SelectedItem.ToString());
        int tienCuoc2 = int.Parse(comboBox2.SelectedItem.ToString());
        if (tienCuoc1 + tienCuoc2 > soTienConLai)
```

```
        {
            MessageBox.Show("Tổng tiền cược không được lớn hơn tổng tiền còn lại!!", "Lưu ý");
            return;
        }
        //groupbox chẵn lẻ
        if (rdChan.Checked)
        {
            if (total % 2 == 0)
                soTienConLai += tienCuoc1;
            else
                soTienConLai -= tienCuoc1;
        }
        else
        {
            if (total % 2 == 0)
                soTienConLai -= tienCuoc1;
            else
                soTienConLai += tienCuoc1;
        }
        //groupbox tổng 3-10 && 11-18
        if (rd3.Checked)
        {
            if (total <= 10)
                soTienConLai += tienCuoc2;
            else //total >= 11
                soTienConLai -= tienCuoc2;

        }
        else
        {
            if (total <= 10)
                soTienConLai -= tienCuoc2;
            else //total >= 11
                soTienConLai += tienCuoc2;
        }
        lbResult.Text = soTienConLai.ToString();
    }

    private void Form1_FormClosing(object sender, FormClosingEventArgs e)
    {
        if (MessageBox.Show("Đóng ứng dụng?", "Thông báo", MessageBoxButtons.YesNo,
MessageBoxIcon.Question) == DialogResult.No)
            e.Cancel = true;
    }

    private void checkBox_CheckedChanged(object sender, EventArgs e)
    {
        pic1.Image = pic2.Image = pic3.Image = Image.FromFile(Application.StartupPath +
@"\HinhXucXac\5.gif");
        pic1.SizeMode = pic2.SizeMode =pic3.SizeMode = PictureBoxSizeMode.Zoom;
        pic1.Visible = pic2.Visible = pic3.Visible = checkBox.Checked;
    }
```

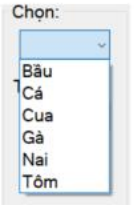
Form đầu tiên: (2,5 đ): Khi chương trình thực thi, xuất hiện một form như sau:

Form không có thanh tiêu đề. Hai hình Bầu, Cua tự động di chuyển vào giữa form, khi chúng gặp nhau thì form trên tự động đóng và xuất hiện cửa sổ chính của chương trình với giao diện như hình bên dưới.

Giao diện form chính (3 đ):

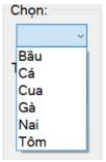


- Form hiển thị giữa màn hình, không cho phép thay đổi kích thước, nút phóng to bị vô hiệu hóa
- Tiêu đề Form hiển thị Mã số - Họ tên sinh viên
- Combobox Chọn hiển thị danh sách như sau:



- **Tiền còn lại** ban đầu hiển thị 1000

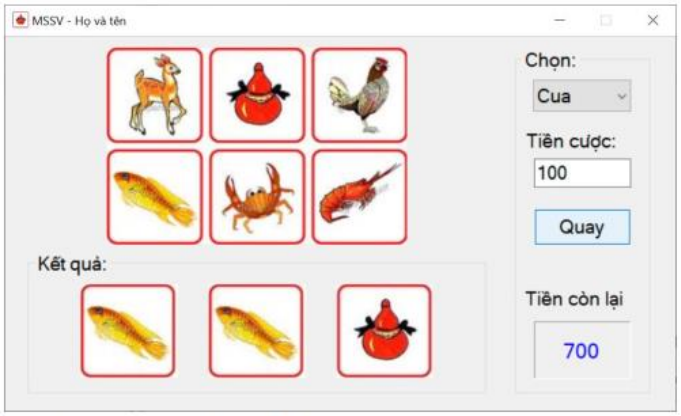
- Form hiển thị giữa màn hình, không cho phép thay đổi kích thước, nút phóng to bị vô hiệu hóa
- Tiêu đề Form hiển thị Mã số - Họ tên sinh viên
- Combobox Chọn hiển thị danh sách như sau:



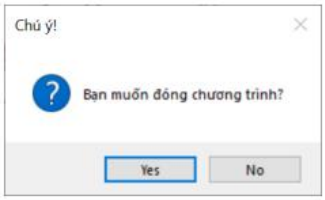
- **Tiền còn lại** ban đầu hiển thị 1000

Ngược lại, người chơi sẽ thua số tiền trên và số tiền còn lại hiển thị trong ô Tiền còn lại trên giao diện. Nếu hết tiền (tiền <=0) thì button Quay bị vô hiệu hóa **(4đ).**

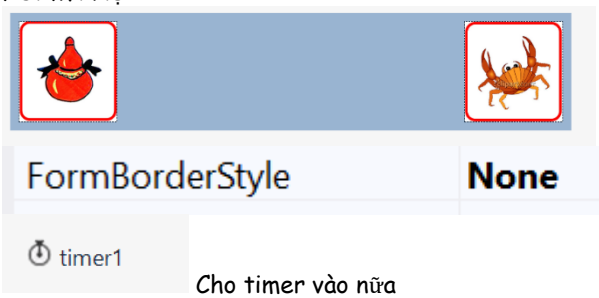
Lưu ý: tiền cược phải là bội số của 100 và không lớn hơn tiền còn lại.



Đóng ứng dụng: hiển thị hộp thoại cảnh báo **(0,5 đ)**



FORM PHỤ



Cho timer vào nữa

```
private void timer1_Tick(object sender, EventArgs e)
    {

        pictureBox1.Location = new Point(pictureBox1.Location.X + 7, pictureBox1.Location.Y);
        pictureBox2.Location = new Point(pictureBox2.Location.X - 7, pictureBox2.Location.Y);
        if (pictureBox1.Bounds.IntersectsWith(pictureBox2.Bounds)) //kiểm tra 2 hình có giao nhau không
        {
            //nếu 2 hình giao nhau thì đóng form và hiển thị form chính
            timer1.Stop();
            this.Hide();
            FormMain f = new FormMain();
            f.Show();

        }
    }

    private void Form1_Load(object sender, EventArgs e)
    {
        timer1.Start();
    }
```

FORM CHÍNH

```
string pathImg = Application.StartupPath;
        int[] result = new int[3];
        int Choose = 0;
        Random rand = new Random();
        int Money = 0;
        int Reward = 0;
        public FormMain()
        {
            InitializeComponent();
        }

        private void btQuay_Click(object sender, EventArgs e)
        {
            Money = Int32.Parse(lbResult.Text);
            Reward = Int32.Parse(txtTienCuoc.Text);
            if (Reward > Money)
            {
                MessageBox.Show("Số tiền cược vượt mức số tiền còn lại!!");
            }
            else
            {
                Index();
            }
        }

        private void FormMain_Load(object sender, EventArgs e)
        {
            pathImg = Application.StartupPath + @"\picBauCua\";
            lbResult.Text = "1000";
        }

        private void Index()
        {

            int Count = 0;
            Choose = comboBox1.SelectedIndex + 1;
            for (int i = 0; i < result.Length; i++)
            {
                result[i] = rand.Next(1, 6);
                if (Choose == result[i])
                {
                    Count++;
                }
            }
            pic1.Image = Image.FromFile(pathImg + result[0].ToString() + ".png");
            pic2.Image = Image.FromFile(pathImg + result[1].ToString() + ".png");
            pic3.Image = Image.FromFile(pathImg + result[2].ToString() + ".png");
            if (Count == 0)
            {

                lbResult.Text = (Money - Reward).ToString();
                MessageBox.Show("Chúc may mắn lần sau!");
            }
            else
            {
```

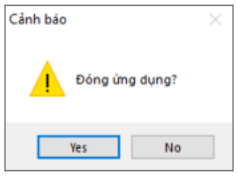
```
        lbResult.Text = (Money + Reward * Count).ToString();
        string reward = "Chúc mừng bạn, có " + Count.ToString() + " " + comboBox1.Text;
        MessageBox.Show(reward);
    }

}
```

- Khi ứng dụng chạy, xuất hiện form sau:

  - Hình ảnh và màu nền, chuỗi tên sinh viên được VẼ (1 đ)
  - Form không có tiêu đề, sau khoảng 3 giây tự động đóng và xuất hiện form chính của ứng dụng (1đ)

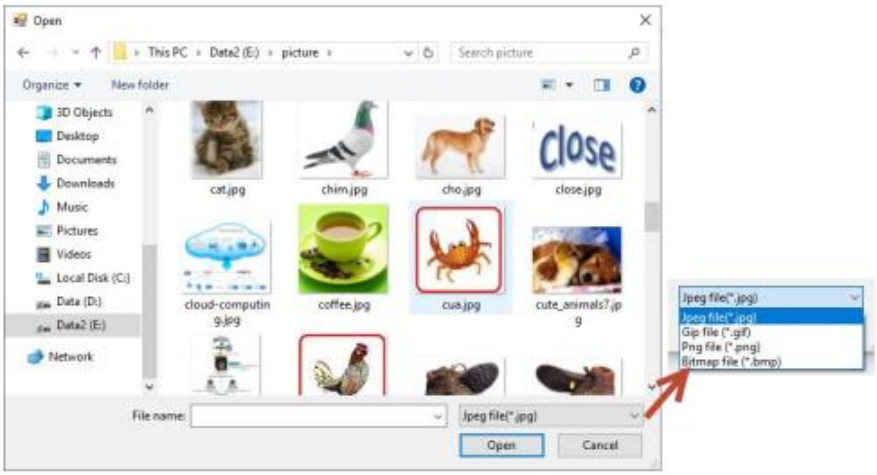
- **Giao diện form chính**: phóng to đầy màn hình, tiêu đề hiển thị mã số sinh viên và họ tên của sinh viên làm bài thi, với menu ban đầu như sau: (0.5 đ)

Menu **File → Exit**: đóng ứng dụng, có hiển thị hộp thoại cảnh báo: (0.5 đ)

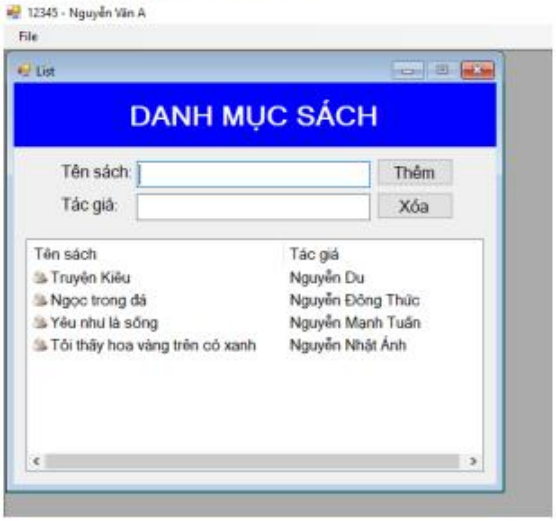


Menu **File → Image**: mở form hiển thị một hình tự động di chuyển về hướng bên trái (1 đ)

- Trên menu File có thêm menu Open (0.5 đ)
- Form không cho phép thay đổi kích thước, nút MaximizeBox bị vô hiệu hóa (0.25 đ)
- Nhấn các phím mũi tên để đổi hướng chuyển động của hình, nếu hình chạy ra khỏi form thì sẽ đi vào từ hướng bên kia. (1 đ)
- Menu File → Open: mở cửa sổ cho phép chọn hình để thay cho hình hiện tại (chỉ hiển thị các tập tin hình theo lựa chọn từ combobox bên dưới) (1 đ)



Menu **File →List**: mở form có giao diện như sau (1 đ)



- Form không cho phép thay đổi kích thước, nút MaximizeBox bị vô hiệu hóa (0.25 đ)
- Danh sách các sách và tên tác giả được tự động hiển thị như hình (0.5 đ)
- Nhập Tên sách, tác giả → nhấn nút **Thêm**, Tên sách và tác giả mới nhập được thêm vào ListView, các ô tên sách và tác giả xóa trống, dấu nháy (focus) tự động đặt vào ô Tên sách. (1 đ)
- Chọn các dòng muốn xóa trong ListView, nhấn nút **Xóa**, các dòng đã chọn sẽ bị xóa khỏi ListView (0.5 đ)

FORM PHỤ

Cho timer

```
public Form2()
    {
       InitializeComponent();

       // Loại bỏ tiêu đề và các cụm nút tối thiểu, tối đa và đóng
       this.FormBorderStyle = FormBorderStyle.None;
       // Căn giữa form trên màn hình
       this.StartPosition = FormStartPosition.CenterScreen;
       // Không hiển thị form trong thanh Taskbar
       this.ShowInTaskbar = false;
       // Luôn nằm trên cùng của các cửa sổ khác
       this.TopMost = true;
       // Đặt đối tượng Timer để đếm thời gian 3 giây
       timer1.Interval = 3000;
       timer1.Start();
    }

    private void timer1_Tick(object sender, EventArgs e)
    {
          timer1.Stop();
          this.Hide();
          Form1 f = new Form1(); //Tạo form chính
          f.Show();

    }

    private void Form2_Paint(object sender, PaintEventArgs e)
    {
       Font f = new Font("Arial", 36, FontStyle.Bold);
       StringFormat fm = new StringFormat();
       Rectangle  rect = new Rectangle(0,0, this.Width, this.Height);
       LinearGradientBrush brush = new LinearGradientBrush(rect, Color.Red, Color.Yellow, 45);
       TextureBrush tbr = new TextureBrush(Image.FromFile(Application.StartupPath + @"\1.jpg"));
       fm.Alignment = StringAlignment.Center;
       fm.LineAlignment = StringAlignment.Center;
       e.Graphics.FillRectangle( brush, rect); //
       e.Graphics.DrawString("Nguyen Van A", f, tbr,ClientRectangle, fm);
    }
```

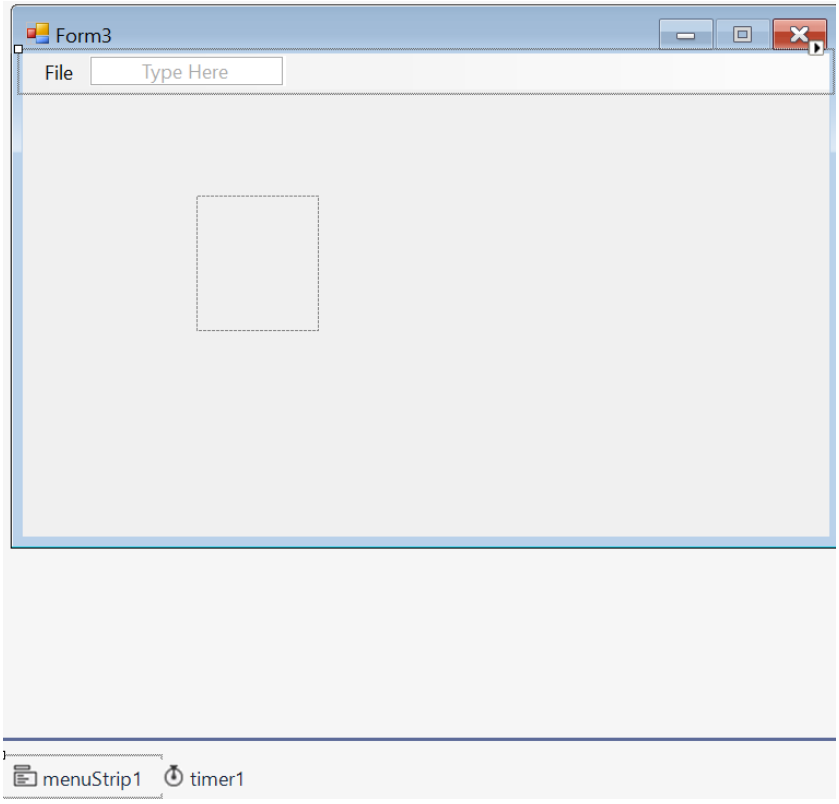
FORM CHÍNH

Tạo menuStrip

```
private void imageToolStripMenuItem_Click(object sender, EventArgs e)
    {
       IsMdiContainer = true;
       //Tạo form con
       Form3 f = new BaiThiThu2.Form3(); //BaiThiThu2 là tên project
       f.MdiParent = this; //Cho nó có thể có nhiều form con
```

```
        f.Show();
    }

    private void MenuList_Click(object sender, EventArgs e)
    {
        List f = new List();
        f.Show();
    }
    //Nút thoát
    private void exitToolStripMenuItem_Click(object sender, EventArgs e)
    {
        DialogResult thongbao;
        thongbao = (MessageBox.Show("Đóng ứng dụng?", "Cảnh báo",
                MessageBoxButtons.YesNo, MessageBoxIcon.Warning));
        if (thongbao == DialogResult.Yes)
            Application.Exit();

    }
```

FORM IMAGE



Tạo menuStrip và timer

menuStrip cho

MergeAction	**MatchOnly**

Chỉnh vị trí của phần từ con lại như đề

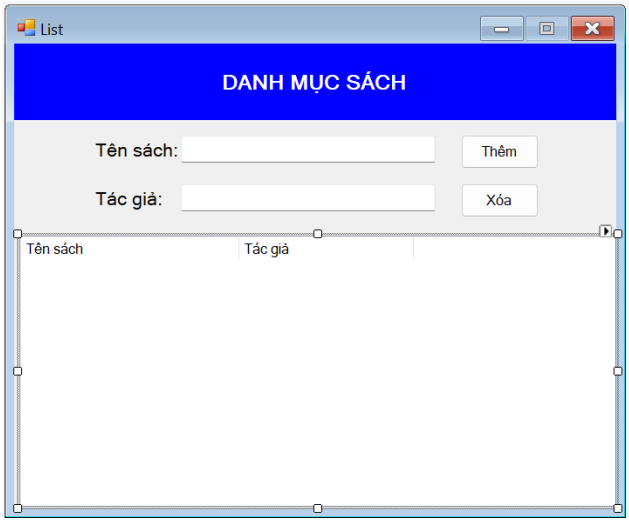
MergeAction	**Insert**
MergeIndex	**1**

```
int dx = 10, dy;
    Image img;
    public Form3()
    {
        InitializeComponent();
    }
```

```
private void Form3_Load(object sender, EventArgs e)
{
   pictureBox1.Image = Image.FromFile(Application.StartupPath + @"\1.jpg");
   timer1.Start();
}

private void LoadImage(String fileName)
{
   img = Image.FromFile(fileName);
   pictureBox1.Image = img;
}

private void openToolStripMenuItem_Click(object sender, EventArgs e)
{
   OpenFileDialog dlg = new OpenFileDialog();
   dlg.Filter = "Bitmap (*.bmp)|*.bmp|" +
                "JPFG (*.jpg)|*.jpg|" +
                "GIF (*.gif)|*.gif|" +
                 "PNG (*.png)|*.png|" +
                 "TIFF (*.tiff)|*.tiff|" +
                 "WMF (*.wmf)|*.wmf";
   if (dlg.ShowDialog() == DialogResult.OK)
   {
      pictureBox1.Image = Image.FromFile(dlg.FileName);
   }
}

private void timer1_Tick(object sender, EventArgs e)
{
   Play();
}
private void Play()
{

   pictureBox1.Left += dx;
   pictureBox1.Top += dy;

   if (pictureBox1.Left >= ClientRectangle.Width)
      pictureBox1.Left = 0; //-pic1.Width

   if (pictureBox1.Right <= 0)
      pictureBox1.Left = ClientRectangle.Width;

   if (pictureBox1.Top >= ClientRectangle.Height)
      pictureBox1.Top = 0; //-pic1.Width%

   if (pictureBox1.Top <= -pictureBox1.Height)
      pictureBox1.Top = ClientRectangle.Height;

}
protected override bool ProcessDialogKey(Keys keyData)
{
```

```
        switch (keyData)
        {
            case Keys.Up: dy = -Math.Abs(10); dx = 0; break;
            case Keys.Down: dy = Math.Abs(10); dx = 0; break;
            case Keys.Left: dx = -Math.Abs(10); dy = 0; break;
            case Keys.Right: dx = Math.Abs(10); dy = 0; break;
        }

        return true;
    }
```

FORM LIST



Có listView

```
 private void button1_Click(object sender, EventArgs e)
    {
        if (textBox1.Text == "" || textBox2.Text == "")
        {
            MessageBox.Show("Bạn cần nhập đầy đủ thông tin");
            return;
        }
        ListViewItem item = new ListViewItem();
        item.Text = textBox1.Text; // Thêm dữ liệu vào cột đầu tiên

        ListViewItem.ListViewSubItem subItem = new ListViewItem.ListViewSubItem();
        subItem.Text = textBox2.Text; // Thêm dữ liệu vào cột thứ hai

        item.SubItems.Add(subItem); // Thêm subItem vào item
        listView1.Items.Add(item); // Thêm item vào listView1
        textBox1.Clear();
        textBox2.Clear();
        textBox1.Focus();


    }
    private void button2_Click(object sender, EventArgs e)
    {
        // Lấy danh sách các mục được chọn
        var selectedItems = listView1.SelectedItems;

        // Xóa từng mục được chọn
        foreach (ListViewItem item in selectedItems)
```

```
    {
        listView1.Items.Remove(item);
    }
}
```