**Một số properties thông dụng**

**MDI**

- Nền xám + menu ⇒ isMDIcontainer = true

**Form:**

- Không phóng to, thu nhỏ: Minisizebox/Maximizebox = false

- Không thay đổi kích thước: FormBorderStyle=fixed..

- Xóa toàn bộ nút control (X,\_): FormBorderStyle=none

-Mặc định loadForm sẽ full màn hình: WindowState = Maximized

**Thuộc tính chung**

- Anchor: controls sẽ dãn theo cạnh được thay đổi kích thước

- Dock: controls sẽ gán vào cạnh được chọn

**Button**

* Thay đổi dạng button: FlatStyle

**TextBox**

* Không cho người dùng tác dụng lên txtbox: ReadOnly = true
* Nhập nhiều dòng: MultiLine = true
* Nhập mật khẩu: passwordChar

**ListView**

* Tương tác được cả dòng trên item: FullRowControl = true
* Kẻ các đường thẳng và ngang: GridLines = true
* Thấy toàn bộ Column: View = Detail
* Có icon khi xem ở dạng Detail: SmallImageList → chèn 1 ImageList

**ListBox**

* SelectionMode: MultiExtended (Chọn nhiều item bằng chuột hoặc ctrl)

**DateTimePicker**

* Thay đổi giao diện: Form = Short/Custom
* Short (mm/dd/yyyy)
* Muốn dạng dmy → format : custom; CustomFormat : dd/MM/yyyy

**FlowLayoutPanel**

* FlowDirection: chuyển hướng

**Radio**

* chuyển radio thành button: Appearance = button

**ToolTip**

* Tạo tooltip tại form
* Chọn control cần xuất hiện tooltip
* Chọn properties: tooltip on tooltip

**Drawing**

* SolidBrush vẽ 1 màu nào đó
* TextureBrush vẽ 1 chữ bằng hình ảnh
* LinearGradientBrush vẽ có hiệu ứng trộn 2 màu
* HatchBrush vẽ có hiệu ứng đặc biệt

**Chương 2:**

**Bài 1: Các phép tính đơn giản**

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

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

{

int a = int.Parse(txtSo1.Text);

int b = int.Parse(txtSo.Text);

lbKetQua.Text = String.Format("{0}", a + b);

}

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

{

int a = int.Parse(txtSo1.Text);

int b = int.Parse(txtSo.Text);

lbKetQua.Text = String.Format("{0}", a - b);

}

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

{

int a = int.Parse(txtSo1.Text);

int b = int.Parse(txtSo.Text);

lbKetQua.Text = String.Format("{0}", a \* b);

}

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

{

int a = int.Parse(txtSo1.Text); // txtSo1, txtSo2 do người dùng nhập

int b = int.Parse(txtSo.Text);

lbKetQua.Text = String.Format("{0:0.00}", (double)a / b); // lấy 2 chữ số thập phân

}

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

{

Close();

}

}

**Bài 2: Các phép tính (Có xử lí ngoại lệ )**

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

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

{

try

{

int a = checked(int.Parse(txtSo1.Text)); //checked để kiểm tra về vấn đề tràn số

int b = checked(int.Parse(txtSo2.Text));

if (rdCong.Checked)

lbKetQua.Text = String.Format("{0}", checked(a + b));

//Tạo group box bên trong có các radio button

// Cac radio button là các phép tính

else if (rdTru.Checked)

lbKetQua.Text = String.Format("{0}", checked(a - b));

else if (rdNhan.Checked)

lbKetQua.Text = String.Format("{0}", checked(a \* b));

else if (rdChia.Checked)

{

if(b==0) throw new DivideByZeroException ("Không thể thực hiện phép chia cho 0!");

//tạo ngoại lệ khi điều kiện thõa

lbKetQua.Text = String.Format("{0:0.00}", (double)a / b);

}

else

lbKetQua.Text = String.Format("{0}", a % b);

}

catch (FormatException)

{

lbKetqua.Text = "Bạn phải nhập hai số!";

}

catch(OverflowException)

{

lbKetqua.Text="Không thể xử lí số quá lớn"

}

catch(DivideByZeroException ex)

{

lbKetqua.Text = ex.Message; // Lấy câu thông báo của trình biên dịch

}

}

}

Bài 3: Viết trò chơi quay số ( Random Number) ( Random 3 số, đoán tổng 3)

public partial class Form1 : Form

{

Random rand = new Random();

int diem = 0;

public Form1()

{

InitializeComponent();

}

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);

lbSo1.Text = so1.ToString();

lbSo2.Text = so1.ToString();

lbSo3.Text = so1.ToString();

if(rd3.Checked)

{

if ((so1 + so2 + so3) <= 10)

diem += 10;

else

diem -= 10;

}

else

{

if ((so1 + so2 + so3) <= 10)

diem -= 10;

else

diem += 10;

}

lbDiem= diem.ToString();

}

}

Bài 4: Bé học toán

public partial class Form1 : Form

{

Random rand = new Random();

string[] arrPhepToan = { "+", "-", "x", ":" };

int vitri;

public Form1()

{

InitializeComponent();

}

private void Form1\_Load(object sender, EventArgs e)

{

TaoPheptoan();

}

private void TaoPheptoan()

{

int a = rand.Next(10); //Sinh số ngẫu nhiên từ 0-9

int b = rand.Next(10);

int vitri=rand.Next(4);

lbSo1.Text = a.ToString();

lbSo2.Text = b.ToString();

lbToantu.Text = arrPhepToan[vitri];//lấy 1 toán tử trong mảng

txtDapan.Text = "";

txtDapan.Focus();//dấu nháy chuột sẽ chỉ tại đây

lbKetqua.Text = "";

}

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

{

TaoPheptoan();

}

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

{

try

{

int a = int.Parse(lbSo1.Text);

int b = int.Parse(lbSo2.Text);

int sochon = int.Parse(txtDapan.Text);

int dapan = 0;

switch (vitri)

{

case 0:

dapan = a + b;

break;

case 1:

dapan = a - b;

break;

case 2:

dapan = a \* b;

break;

case 3:

dapan = a / b;

break;

}

if (sochon == dapan)

lbKetqua.Text() = "Đúng rồi";

else

lbKetqua.Text() = "Sai! Kết quả là: " + dapan.ToString();

}

catch { }

}

private void bt0\_Click(object sender, EventArgs e)

{

Button bt = (Button)sender; // Ctrl chọn các nút còn lại rồi bắt sự kiện Click, chọn bt0\_Click

txtDapan.Text += bt.Text;

}

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

{

if (txtDapan.Text != "")

return;

txtDapan.Text = "-";

}

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

{

if (txtDapan.Text == "") return;

txtDapan.Text = txtDapan.Text.Substring(0, txtDapan.Text.Length - 1);

}

}

**Bài 5: Bài xử lý chuỗi**

public partial class Form1 : Form

{

string strText = "khoa Công nghệ thông tin, đại học MỞ THÀNH phố HỒ CHÍ MINH";

public Form1()

{

InitializeComponent();

}

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

{

int pos = int.Parse(txtInsertPos.Text);

if (pos < 0 || pos > txtSo1.Text.Length)

return;

txtSo1.Text = txtSo1.Text.Insert(pos, txtSo2.Text);

}

private void Form1\_Load(object sender, EventArgs e)

{

labelName.Text = " CHƯƠNG TRÌNH MINH HOẠ XỬ LÝ CHUỖI ";

txtSo1.Text = strText;

}

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

{

txtSo1.Text = txtSo1.Text.Replace(txtSo2.Text, txtSo3.Text);

}

private void btDeleteS2\_Click(object sender, EventArgs e)

{

int pos = txtSo1.Text.IndexOf(txtSo2.Text);

while(pos>=0)

{

txtSo1.Text= txtSo1.Text.Remove(pos,txtSo2.Text.Length);

pos=txtSo1.Text.IndexOf(txtSo2.Text);

}

}

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

{

txtSo1.Text = strText;

}

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

{

int pos = int.Parse(txtStartPos.Text);

int numchar = int.Parse(txtNumChar.Text);

txtSo2.Text = txtSo1.Text.Substring(pos, numchar);

}

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

{

char[] token = {' ' , '\t'};

// Tách từ trong s1 vào mảng arr theo các kí tự trong mảng token

string[] arr = txtSo1.Text.Split(token,StringSplitOptions.RemoveEmptyEntries);

//Đảo mảng

Array.Reverse(arr);

txtSo1.Text = String.Join(" ", arr); // chuyển mảng thành chuỗi

}

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

{

char[] token = { ' ', '\t' };

string[] arr = txtSo1.Text.Split(token, StringSplitOptions.RemoveEmptyEntries);

// duyệt từng phần tử trong mảng, chuyển hoa kí tự đầu, chuyển thường các kí tự còn lại

for(int i=0; i <arr.Length; i++)

{

arr[i] = arr[i].Substring(0,1).ToUpper() + arr[i].Substring(1).ToLower();

}

txtSo1.Text=String.Join(" ", arr);

}

private void timer1\_Tick(object sender, EventArgs e)

{

labelName.Text = labelName.Text.Substring(labelName.Text.Length - 1) + labelName.Text.Substring(0, labelName.Text.Length - 1); //trái sang phải

}

}

**Chương 3:**

**Form**

**Phân số:**

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

{

try

{

Button bt = (Button)sender;

PS ps1 = new PS(Int32.Parse(txtTS1.Text), Int32.Parse(txtMS1.Text));

PS ps2 = new PS(Int32.Parse(txtTS2.Text), Int32.Parse(txtMS2.Text));

if (bt == btCong)

{

PS kq = ps1.Cong(ps2);

txtTSKQ.Text = kq.TuSo.ToString();

txtMSKQ.Text = kq.Mauso.ToString();

lbPhepTinh.Text = bt.Text;

}

else if (bt == btTru)

{

PS kq = ps1.Tru(ps2);

txtTSKQ.Text = kq.TuSo.ToString();

txtMSKQ.Text = kq.Mauso.ToString();

lbPhepTinh.Text = bt.Text;

}

else if (bt == btNhan)

{

PS kq = ps1.Nhan(ps2);

txtTSKQ.Text = kq.TuSo.ToString();

txtMSKQ.Text = kq.Mauso.ToString();

lbPhepTinh.Text = bt.Text;

}

else if (bt == btChia)

{

PS kq = ps1.Chia(ps2);

txtTSKQ.Text = kq.TuSo.ToString();

txtMSKQ.Text = kq.Mauso.ToString();

lbPhepTinh.Text = bt.Text;

}

}

catch (Exception ex)

{

MessageBox.Show(ex.Message.ToString());

}

}

**Class Phanso**

internal class PS

{

int tuso, mauso;

public PS()

{

tuso = 0;

mauso = 1;

}

public PS(int tuso, int ms)

{

this.tuso = tuso;

this.mauso = ms == 0 ? 1 : ms;//neu nhap mau so bang 0 thi chuyen sang 1

}

//tao cac property

public int TuSo

{

get { return tuso; }

set { tuso = value; }

}

public int Mauso

{

get { return mauso; }

set { mauso = value; }

}

//cac phuong thuc tinh toan

public PS Cong(PS p)

{

PS kq = new PS();

if (this.mauso == p.mauso)

{

kq.tuso = this.tuso + p.tuso;

kq.mauso = this.mauso;

}

else

{

kq.mauso = p.mauso \* this.mauso;

kq.tuso = this.tuso \* p.mauso + p.tuso \* this.mauso;

}

kq.ToiGian();

return kq;

}

public PS Tru(PS p)

{

PS kq = new PS();

if (this.mauso == p.mauso)

{

kq.tuso = this.tuso - p.tuso;

kq.mauso = this.mauso;

}

else

{

kq.tuso = this.tuso \* p.mauso - p.tuso \* this.mauso;

kq.mauso = this.mauso \* p.mauso;

}

kq.ToiGian();

return kq;

}

public PS Nhan(PS p)

{

PS kq = new PS();

kq.tuso = this.tuso \* p.tuso;

kq.mauso = this.mauso \* p.mauso;

kq.ToiGian();

return kq;

}

public PS Chia(PS p)

{

PS kq = new PS();

kq.tuso = this.tuso \* p.mauso;

kq.mauso = this.mauso \* p.tuso;

kq.ToiGian();

return kq;

}

public void ToiGian()

{

int uc = LopChung.UCLN(tuso, mauso);

if (uc > 0)

{

this.tuso=this.tuso/uc;

this.mauso = this.mauso / uc;

}

}

**Class LopChung**

internal class LopChung

{

public static int UCLN(int a, int b)

{

int a1 = Math.Abs(a);

int b1 = Math.Abs(b);

if (b1 == 0)

{

return a1;

}

return UCLN(b1, a1%b1);

}

}

**Chương 4:**

**Ball tự động di chuyển**

int dx=10, dy=10;

private void timer1\_Tick(object sender, EventArgs e)

{

if(pictureBox1.Left < 0 || pictureBox1.Right > ClientRectangle.Width)

{

dx = -dx; // chạm 2 cạnh trái/phải thì quay lại

}

if (pictureBox1.Top < 0 || pictureBox1.Bottom > ClientRectangle.Height)

{

dy = -dy; // chạm 2 cạnh trên/dưới thì quay lại

}

pictureBox1.Left += dx; // tự động di chuyển sang phải

pictureBox1.Top += dy; // tự động di chuyển xuống

}

**Xúc xắc**

string pathImg;

int nCount;

int nLose;

int nWin;

int nChoose;

Random rand = new Random();

private void Form1\_Load\_1(object sender, EventArgs e)

{

Init();

}

private void Init()

{

pathImg= Application.StartupPath+@"\HinhXucXac\";

nChoose= 1;

nWin= 0;

nLose = 0;

nCount = 0;

picChon.Image = Image.FromFile(pathImg+"dice1.png");

picKQ.Image = null;

lbCount.Text = "Số lần quay: "+ nCount;

lbWin.Text = String.Format("Số lần thắng: {0} ({1:0.00})",nWin,(double)nWin\*100/nCount);

lbLose.Text = String.Format("Số lần thua: {0} ({1:0.00})", nLose, (double)nLose \* 100 / nCount);

}

private void Form1\_Click(object sender, EventArgs e)

{

Button bt = (Button) sender;

picChon.Image = Image.FromFile(pathImg + "dice" + bt.Text + ".png");

nChoose = Int32.Parse(bt.Text);

}

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

{

int n = rand.Next(1, 7);

picKQ.Image = Image.FromFile(pathImg + "dice" + n + ".png");

nCount++;

if(nChoose==n)//doan dung

{

nWin++;

listKQ.Items.Add("Thắng ( đoán " + nChoose + " ra " + n+")");

}

else

{

nLose++;

listKQ.Items.Add("Thua ( đoán " + nChoose + " ra " + n+")");

}

lbCount.Text = "Số lần quay: " + nCount;

lbWin.Text = String.Format("Số lần thắng: {0} ({1:0.00})", nWin, (double)nWin \* 100 / nCount);

lbLose.Text = String.Format("Số lần thua: {0} ({1:0.00})", nLose, (double)nLose \* 100 / nCount);

}

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

{

Init();

listKQ.Items.Clear();

}

protected override bool ProcessDialogKey(Keys keyData)

{

if (keyData == Keys.Enter)

{

int n = rand.Next(1, 7);

picKQ.Image = Image.FromFile(pathImg + "dice" + n + ".png");

nCount++;

if (nChoose == n)//doan dung

{

nWin++;

listKQ.Items.Add("Thắng ( đoán " + nChoose + " ra " + n + ")");

}

else

{

nLose++;

listKQ.Items.Add("Thua ( đoán " + nChoose + " ra " + n + ")");

}

lbCount.Text = "Số lần quay: " + nCount;

lbWin.Text = String.Format("Số lần thắng: {0} ({1:0.00})", nWin, (double)nWin \* 100 / nCount);

lbLose.Text = String.Format("Số lần thua: {0} ({1:0.00})", nLose, (double)nLose \* 100 / nCount);

}

else if(keyData == Keys.Escape)

{

Init();

listKQ.Items.Clear();

}

return base.ProcessDialogKey(keyData);

}

**Quản lí nhân viên**

* **Đăng nhập**

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

{

try

{

if(txtMatKhau.Text != "admin")//kiểm tra mật khẩu đúng ko

{

MessageBox.Show("Sai mật khẩu");

}

else

{

Form4.Ten = txtTen.Text;//lấy dữ liệu từ formDangNhap sang form Chính

this.Close();

}

}

catch(Exception ex)

{

MessageBox.Show(ex.Message);

}

}

private void DangNhap\_FormClosing(object sender, FormClosingEventArgs e)

{

if(Form4.Ten == "") //nếu dữ liệu form chính rỗng thì thoát

{

Application.Exit();

}

}

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

{

Close();

}

* **Form Chính**

public bool bClose;

public static string Ten = ""; **//Phải là public static mới lấy dữ liệu được từ 1 form khác**

public Form4()

{

DangNhap dangNhap = new DangNhap(); //hiện form DangNhap trước khi formChinh hiện

dangNhap.ShowDialog();//chỉ mở được Form khác khi form DangNhap tắt (do ShowDialog)

InitializeComponent();

}

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

{

* Các bước thêm 1 item vào ListView

0. Tạo ListView **(nếu chưa tạo ListView trên Form)**

ListView listView1 = new ListView();

1. Tạo 1 đối tượng ListViewItem

ListViewItem listItem = new ListViewItem(txtTen.Text);//tham số truyền

vào luôn là giá trị cần điền ở column đầu tiên

1. Add các giá trị tiếp theo vào đối tượng ListViewItem bằng SubItems.Add

listItem.SubItems.Add(dateTime.Text);

if(rdNam.Checked)

{

listItem.SubItems.Add(rdNam.Text);

listItem.ImageIndex = 0;

}

else

{

listItem.SubItems.Add(rdNu.Text);

listItem.ImageIndex = 1;

}

1. Add đối tượng ListViewItem vào ListView

listView1.Items.Add(listItem);

1. Add ListView vào Form **(chỉ làm khi có làm bước 0)**

this.Controls.Add(listView1);

}

private void Form4\_Load(object sender, EventArgs e)

{

lbTenDangNhap.Text = Ten; //lấy dữ liệu từ form DangNhap

bClose = false;

}

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

{

//xóa toàn bộ phần tử được chọn

foreach(ListViewItem item in listView1.SelectedItems)

{

listView1.Items.Remove(item);//xóa từng phần tử được chọn

}

}

private void timer1\_Tick(object sender, EventArgs e)

{

//hiệu ứng close từ từ

this.Opacity - = 0.05; //độ rõ của form giảm từ từ

if(this.Opacity <= 0)

{

bClose=true;

Application.Exit();

}

}

private void Form4\_FormClosing(object sender, FormClosingEventArgs e)

{

if(!bClose)

{

e.Cancel = true;

timer1.Enabled = true; //chạy timer\_tick (set mặc định trên form Enable = false)

}

}

**Chương 5:**

**Di chuyển hình vào trong LayoutPanel**

Random rd = new Random();

Point point = new Point(); //lấy vị trí trước khi Mouse\_Move của hình

int count = 0;

string[] arrFiles;// chứa toàn bộ đường dẫn file

private void Form1\_Load(object sender, EventArgs e)

{

label1.Text = "Dùng chuột hoặc các phím mũi tên di chuyển ảnh vào trong khung bên phải";

arrFiles = Directory.GetFiles(Application.StartupPath + @"\image");//lấy tất cả ảnh có trong đường dẫn

AddNewPicture();

}

private void AddNewPicture()

{

* Cách tạo 1 pictureBox chưa có trên form tương tự ListView

count++; // để đặt tên cho pictureBox vừa tạo

string file = arrFiles[rd.Next(arrFiles.Length)]; // lấy ngẫu nhiên đường dẫn trong mảng

Image image = Image.FromFile(file);

PictureBox pictureBox = new PictureBox();//tạo 1 pictureBox

pictureBox.Image = image;

pictureBox.Name=count.ToString(); //đặt tên PictureBox bằng biến count

pictureBox.Location = new Point(0, 0); //PictureBox được tạo tại vị trí (0,0)

pictureBox.SizeMode = PictureBoxSizeMode.StretchImage;

pictureBox.Width = pictureBox.Height = 100;

this.Controls.Add(pictureBox); **// phải add pictureBox vào Form nếu ko có sẽ ko có ảnh**

this.Controls.SetChildIndex(flowLayoutPanel1, this.Controls.Count - 1);//set độ ưu tiên nằm trên Controls khác (nếu không set cái nào tạo ra sau sẽ nằm dưới)

pictureBox.MouseDown += PictureBox\_MouseDown;

pictureBox.MouseMove += PictureBox\_MouseMove;

pictureBox.MouseUp += PictureBox\_MouseUp;

}

private void PictureBox\_MouseUp(object sender, MouseEventArgs e)

{

PictureBox picture = sender as PictureBox; //tương đồng với ép kiểu (PictureBox) sender

if (flowLayoutPanel1.Bounds.Contains(picture.Bounds)) // nếu pictureBox nằm trong layoutPanel thì thêm vào

{

flowLayoutPanel1.Controls.Add(picture);

AddNewPicture();

}

}

private void PictureBox\_MouseMove(object sender, MouseEventArgs e)

{

if(e.Button==MouseButtons.Left) //nếu nhấn chuột trái (nhấn giữ)

{

PictureBox picture = (PictureBox)sender;

//di chuyển picBox đến vị trí mới

int dx = e.X - point.X;

int dy = e.Y - point.Y;

picture.Left += dx;

picture.Top += dy;

}

}

private void PictureBox\_MouseDown(object sender, MouseEventArgs e)

{

point = e.Location; //lấy tại vị trí nhấn chuột

}

private void Form1\_KeyDown(object sender, KeyEventArgs e)

{

Control[] arrControl = this.Controls.Find(count.ToString(), false);//tìm tất cả các control có tên cần tìm (dù chỉ có 1 phần tử vẫn phải để mảng vì Find trả về kiểu mảng)

PictureBox pictureBox = (PictureBox)arrControl[0];

switch (e.KeyCode)

{

//di chuyển ảnh

case Keys.Left:

if ((pictureBox.Left-5) > 0) pictureBox.Left-=5;

break;

case Keys.Right:

if ((pictureBox.Right+5) < ClientRectangle.Width) pictureBox.Left+=5;

break;

case Keys.Up:

if ((pictureBox.Top-5) > 0) pictureBox.Top-=5;

break;

case Keys.Down:

if ((pictureBox.Bottom+5) < ClientRectangle.Height) pictureBox.Top+=5;

break;

}

if (flowLayoutPanel1.Bounds.Contains(pictureBox.Bounds))

{

flowLayoutPanel1.Controls.Add(pictureBox);

AddNewPicture();

}

}

**Chương 6:**

**PhotoAlbum**

namespace TreeView

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void Form1\_Load(object sender, EventArgs e)

{

InitTree(); // Khởi tạo cây mới

}

void InitTree()

{

string[] arrDrives = Directory.GetLogicalDrives(); //lấy danh sách các ổ đĩa lưu vào mảng arrDrives

TreeNode node = null; //tạo node mới

foreach(string drive in arrDrives) //duyệt từng ổ đĩa

{

node = new TreeNode(drive); gán tên ổ đĩa cho node

node.ImageIndex = 1; //chỉnh ảnh có index = 1 cho node ổ đĩa

node.SelectedImageIndex = 1; //chỉnh image có index = 1 khi chọn node

node.Nodes.Add("temp"); //gán node tạm cho node ổ đĩa

treeView1.Nodes.Add(node);thêm node ổ đĩa vào cây

}

}

//sự kiện trước khi mở rộng folder

private void treeView1\_BeforeExpand(object sender, TreeViewCancelEventArgs e)

{

TreeNode curNode = e.Node; // tạo Node tên curnode để lưu node của ổ đĩa

curNode.Nodes.Clear(); //Xóa các nút con tạm của ổ đĩa

TreeNode node = null; //tạo node mới để lưu các node con của ổ đĩa

try

{

string [] arrDir = Directory.GetDirectories(curNode.FullPath); //trả về danh sách các thư mục con

foreach(string dir in arrDir) //duyệt qua từng phần từ trong ds thư mục con

{

node = new TreeNode(Path.GetFileName(dir)); //gán tên của thư mục con cho node

node.ImageIndex = 0; //Chỉnh hình có index =0 cho node

node.Nodes.Add("tmp"); // thêm nốt con tạm cho node

curNode.Nodes.Add(node); //thêm node vào cây TreeView

}

}

catch

{

}

}

private void treeView1\_AfterExpand(object sender, TreeViewEventArgs e) // sau khi mở rộng thư mục

{

if (e.Node.Level > 0) //nếu node dc mở không phải là ổ đĩa

{

e.Node.ImageIndex = 2; gán hình ảnh có index = 2 cho node

e.Node.SelectedImageIndex = 2; gán hình ảnh có index = 2 cho node dc chọn

}

}

private void treeView1\_AfterCollapse(object sender, TreeViewEventArgs e)

//sau khi đóng thư mục

{

if (e.Node.Level > 0) // nếu không phải ổ dĩa

{

e.Node.ImageIndex = 0; //gán hình ảnh index = 0 cho node

e.Node.SelectedImageIndex = 0; //gán hình ảnh index = 0 cho node dc chọn

}

}

private void treeView1\_AfterSelect(object sender, TreeViewEventArgs e)

//sau khi chọn folder

{

try

{

flowLayoutPanel1.Controls.Clear(); // Dọn sạch flowLayoutPanel

string[] arrFiles = Directory.GetFiles(e.Node.FullPath); // lấy các file con của folder được chọn

foreach(string file in arrFiles) //duyệt từng thành phần trong mảng file

{

//kiểm tra nếu đúng định dạng hình ảnh

if(file.ToLower().EndsWith(".jpg") ||

file.ToLower().EndsWith(".png") ||

file.ToLower().EndsWith("bmp") ||

file.ToLower().EndsWith("gif")

)

{

PictureBox pic = new PictureBox(); //tạo ra một PictureBox pic

pic.Image = Image.FromFile(file); lấy ảnh của file đưa vào pic

pic.Height = flowLayoutPanel1.Height - 5; //chỉnh chiều cao

pic.Width = pic.Height - 5; //chiều dài

pic.SizeMode = PictureBoxSizeMode.StretchImage; //chỉnh chế độ kích thước

pic.Click += Pic\_Click; thêm sự kiện khi click lên pictureBox

flowLayoutPanel1.Controls.Add(pic); //thêm pic vào flowlayoutPanel

}

}

}

catch

{

}

}

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

{

PictureBox pic = (PictureBox)sender; //lấy đối tượng ảnh được click

pB1.Image = pic.Image; đưa lên pB1 ⇒ đưa lên pictureBox to

pB1.SizeMode = PictureBoxSizeMode.StretchImage; //chỉnh chế độ kích thước của hình ảnh

}

}

}

**EmployeeManager**

**Form Main**

using System.Runtime.Serialization;//khai báo thư viện để lưu file

using System.Runtime.Serialization.Formatters.Binary;//khai báo thư viện để chuyển file sang định dạng nhị phân

using System.IO;

namespace Employee\_Manager

{

public partial class Form1 : Form

{

ArrayList arrEmp = new ArrayList();

public Form1()

{

InitializeComponent();

}

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

{

if(txtName.Text == "")

{

errorProvider1.SetError(txtName,"Vui lòng nhập tên!");//thêm Error Provider sẽ hiện lỗi bên

//cạnh đối tượng điều khiển đc chọn

txtName.Focus();//Focus

return;//thoát khỏi hàm

}

Empl empl = new Empl(txtName.Text,Int32.Parse(txtAge.Text),txtAddress.Text);//Khai báo đối tượng

//class Empl để lưu

//các giá trị trong các textbox

arrEmp.Add(empl);//thêm đối tượng empl vào mảng arrEmp chờ để chuyển qua list

txtName.Text = "";//chuyển txtName về ""

txtAge.Text = "";//chuyển txtAge về ""

txtAddress.Text = "";//chuyển txtAddress về ""

txtName.Focus();//chuyển focus()

}

private void txtAge\_KeyPress(object sender, KeyPressEventArgs e)//lấy sự kiện ấn phím trên txtAge

{

if(!char.IsDigit(e.KeyChar) && !char.IsControl(e.KeyChar))//nếu ko ấn phím số

{

e.Handled = true;//bỏ qua bước ấn hiển thị phím dc ấn lên txtAge

}

}

private void tabControl1\_SelectedIndexChanged(object sender, EventArgs e)//sự kiện chuyển tab trong tab control

{

if (tabControl1.SelectedIndex == 1)//nếu tab dc chọn là 1

{

listEmpl.Items.Clear();//xóa các thành phần cũ

foreach (Empl a in arrEmp)//duyệt từng phần tử trong mảng arr

{

ListViewItem item = new ListViewItem(a.Name);//khai báo biến listView có item a.Name

item.SubItems.Add(a.Age.ToString());//thêm đối tượng con của item là a.Age

item.SubItems.Add(a.Address);//thêm đối tượng con của item là a.address

listEmpl.Items.Add(item);//thêm đối tượng item vào listView

}

}

}

private void saveToolStripMenuItem\_Click(object sender, EventArgs e)//bắt sự kiện ấn nút lưu file

{

SaveFileDialog savedl = new SaveFileDialog();//khai báo đối tượng savefiledialog

savedl.Filter = "Empl file| \*.empl";//định dạng file lưu

if(savedl.ShowDialog() == DialogResult.OK)//nếu người dùng chọn OK

{

FileStream stream = new FileStream(savedl.FileName, FileMode.Create, FileAccess.Write);

//đối tượng FileStream để lưu dữ liệu

//+savedl.FileName: là tên tệp tin mà người dùng chọn để lưu đối tượng

//+FileMode: là chế độ tạo tập tin: Create=>lưu/Open=>mở

//+FileAccess: Là quyền truy cập: Write=>lưu/Read=>mở

BinaryFormatter bf = new BinaryFormatter();//đối tượng chuyển dữ liệu sang nhị phân

foreach (Empl em in arrEmp)//duyệt từng phần tử trong arrEmp

{

bf.Serialize(stream, em);//ghi dữ liệu của đối tượng em vào luồng stream

}

stream.Close();//đóng luồng dữ liệu

}

return;

}

private void openToolStripMenuItem\_Click(object sender, EventArgs e)//bắt sự kiện ấn nút mở file

{

LoadData();//thực hiện hàm loadData

tabControl1.SelectedIndex = 1;//sau khi lấy dữ liệu thì chuyển sang bảng thứ 2

}

private void LoadData()//thực hiện lấy data từ file empl

{

OpenFileDialog opendl = new OpenFileDialog();//khởi tạo đối tượng saveFileDialog

opendl.Filter = "Empl file|\*.empl";//Thiết lập định dạng cho File

if (opendl.ShowDialog() == DialogResult.OK)//nếu người dùng chọn OK

{

arrEmp.Clear();//xóa những đối tượng đang có trong arrEmp

FileStream stream = new FileStream(opendl.FileName, FileMode.Open, FileAccess.Read);

//tạo đối tượng luồng lưu file

BinaryFormatter bf = new BinaryFormatter();//chuyển đối tượng sang nhị phân

while (true)//nếu vẫn còn dữ liệu trong file

{

try

{

Empl em = (Empl)bf.Deserialize(stream);//Phục hồi kiểu dữ liệu binary sang kiểu empl

arrEmp.Add(em);//thêm đối tượng vừa chuyển đổi sang em

}

catch (SerializationException ex)//nếu có lỗi khi đọc file

{

break;//thoát khỏi hàm while

}

}

stream.Close();//đóng luồng dữ liệu

}

return;//thoát hàm

}

}

}

**Class EMPL**

namespace EmployeeManager

{

[Serializable] // cần thiết để lưu và lấy dữ liệu từ file

class Empl

{

// Khai báo các biến

private string name;

private int age;

private string address;

public Empl(string name, int age, string address) //hàm khởi tạo

{

this.name = name;

this.age = age;

this.address = address;

}

public string Name

{

get { return this.name; }

set { this.name = value; }

}

public int Age

{

get { return this.age; }

set { this.age = value; }

}

public string Address

{

get { return this.address; }

set { this.address = value; }

}

}

}

**Chương 7:**

**FORM MAIN**

namespace SimplePaint

{

public partial class FrmMain : Form

{

public FrmMain()

{

InitializeComponent();

}

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

{

Application.Exit();

}

private void menuDrawText\_Click(object sender, EventArgs e) //gọi form vẽ chữ

{

FrmDrawText f = new FrmDrawText();

f.MdiParent = this;

f.Show();

}

private void menuDrawByMouse\_Click(object sender, EventArgs e) //gọi form vẽ bằng chuột

{

FrmDrawWithMouse f = new FrmDrawWithMouse();

f.MdiParent = this;

f.Show();

}

private void drawImageToolStripMenuItem\_Click(object sender, EventArgs e) //gọi form vẽ hình ảnh

{

FrmDrawImage f = new FrmDrawImage();

f.MdiParent = this;

f.Show();

}

}

}

**DRAW WITH MOUSE**

using System.Drawing.Drawing2D;

namespace SimplePaint

{

public partial class FrmDrawWithMouse : Form

{

Point POld; //tạo biến lưu vị trí cũ của chuột

Color color;

int penWidth;

Bitmap bmpTemp; //tạo biến bitmap

public FrmDrawWithMouse()

{

InitializeComponent();

}

private void FrmDrawWithMouse\_MouseDown(object sender, MouseEventArgs e)

{

POld = e.Location; lưu lại vị trí của con trỏ khi ấn xuống

}

private void FrmDrawWithMouse\_MouseMove(object sender, MouseEventArgs e)

{

if(e.Button == MouseButtons.Left) //nếu ấn bằng chuột trái

{

Pen pen = new Pen(color,penWidth); tạo đối tượng bút

pen.StartCap = LineCap.Round; //đầu nét vẽ là hình tròn

pen.EndCap = LineCap.Round; //cuối nét vẽ là hình tròn

Graphics g = Graphics.FromImage(bmpTemp); //tạo đối tượng graphic để vẽ trên bitmap

g.DrawLine(pen, POld, e.Location);

POld = e.Location; //cập nhật điểm pOld của bút

Invalidate(); yêu cầu form vẽ lại

}

}

private void FrmDrawWithMouse\_Load(object sender, EventArgs e)

{

color = Color.Red; //cập nhật màu

penWidth = 1; //cập nhật độ dày nét vẽ

bmpTemp = new Bitmap(Screen.PrimaryScreen.Bounds.Width, Screen.PrimaryScreen.Bounds.Height); // set chiều cao, và chiều ngang của bitmap

}

private void FrmDrawWithMouse\_Paint(object sender, PaintEventArgs e)

{

e.Graphics.DrawImage(bmpTemp, 0, 0); //vẽ hình ảnh là bitmap đã được vẽ bằng chuột trước đó

}

protected override bool ProcessDialogKey(Keys keyData) //bắt sự kiện nhất phím

{

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:

if (penWidth < 50) //nếu nét vẽ nhỏ hơn 50

{

penWidth++; //tăng kích thước nét vẽ

}

break;

case Keys.Down:

if (penWidth > 1) //nếu kích thước nét vẽ >1

{

penWidth--; //giảm kích thước nét vẽ

}

break;

}

return true;

}

}

}

**DRAW TEXT**

using System.Drawing.Drawing2D;

namespace SimplePaint

{

public partial class FrmDrawText : Form

{

public FrmDrawText()

{

InitializeComponent();

}

private void FrmDrawText\_Paint(object sender, PaintEventArgs e)

{

Font font = new Font("Arial", 64, FontStyle.Bold); //tạo đối tượng font

SolidBrush br = new SolidBrush(Color.Green); tạo một đối tượng solidBrush // ảnh đơn màu

StringFormat fm = new StringFormat(); tạo đối tượng định dạng cho chuỗi

fm.Alignment = StringAlignment.Far; chỉnh lề chuỗi qua bên phải

e.Graphics.DrawString("HELLO", font, br,ClientRectangle,fm); //vẽ chữ

//=========================

Image img = Image.FromFile("Anh.jpg");

//tạo đối tượng hình ảnh và gán đường dẫn của ảnh cho đối tượng

fm.Alignment = StringAlignment.Near; //chỉnh lề qua phải

fm.LineAlignment = StringAlignment.Far;//chỉnh lề cao của chữ xuống thấp

TextureBrush tbr = new TextureBrush(img); //tạo một đối tượng texturebrush //vẽ bằng hình image

e.Graphics.DrawString("HELLO", font, tbr, ClientRectangle, fm); //Vẽ chữ

//=========================

HatchBrush hbr = new HatchBrush(HatchStyle.DarkVertical,Color.Red,Color.Yellow); //ảnh kiểu này kiểu kia

//tạo đối tượng cọ vẽ HatchBrush

fm.LineAlignment = StringAlignment.Near; //Chỉnh lề thành cao

fm.FormatFlags = StringFormatFlags.DirectionVertical; //Chỉnh văn bản được vẽ theo hướng dọc

e.Graphics.DrawString("HELLO", font, hbr, ClientRectangle, fm);//Vẽ chữ

//=========================

LinearGradientBrush lbr = new LinearGradientBrush(new Rectangle(0,0,30,30),Color.Blue,Color.White,45); //tạo đối tượng cọ vẽ linearGradient //chữ vẽ có màu 45 độ

fm.Alignment = StringAlignment.Far; //Chỉnh lề chữ qua bên phải

fm.LineAlignment = StringAlignment.Far; // chỉnh lề chữ xuống thấp

e.Graphics.DrawString("HELLO", font, lbr, ClientRectangle, fm); //vẽ chữ

}

//nếu như thay đổi kích thước của form thì yêu cầu vẽ lại

private void FrmDrawText\_SizeChanged(object sender, EventArgs e)

{

Invalidate();

}

}

}

**DRAW IMAGE**

namespace SimplePaint

{

public partial class FrmDrawImage : Form

{

public FrmDrawImage()

{

InitializeComponent();

}

private void FrmDrawImage\_Paint(object sender, PaintEventArgs e)

{

Rectangle rc1 = new Rectangle(0, 0, ClientRectangle.Width/2,ClientRectangle.Height/2); //tạo đối tượng rectangle rc1

Rectangle rc2 = new Rectangle(0, ClientRectangle.Height / 2, ClientRectangle.Width / 2, ClientRectangle.Height / 2); //tạo đối tượng rectangle rc2

Rectangle rc3 = new Rectangle(ClientRectangle.Width / 2, 0, ClientRectangle.Width / 2, ClientRectangle.Height); //tạo đối tượng rectangle rc3

DrawImage(rc1, e.Graphics); //gọi hàm vẽ hình

DrawText(rc2, e.Graphics); // Gọi hàm vẽ chữ

DrawPolygon(rc3, e.Graphics); //Gọi hàm vẽ đa giác

Pen pen = new Pen(Color.Blue, 3); //khởi tạo đối tượng bút

e.Graphics.DrawRectangles(pen, new Rectangle[] { rc1, rc2, rc3 }); //Vẽ các hình chữ nhật có trong mảng

}

private void FrmDrawImage\_SizeChanged(object sender, EventArgs e)

{

Invalidate(); //Vẽ lại khi kích thước của sổ thay đổi

}

//================================

void DrawImage(Rectangle rc, Graphics g) //hàm vẽ hình ảnh

{

Image img = Image.FromFile("D:\\SimplePaint\\SimplePaint\\bin\\Debug\\meo.jpg");

//tạo đối tượng ảnh mới

g.DrawImage(img, rc);//vẽ hình ảnh

Color color = Color.FromArgb(70, 255, 255, 0); //tạo đối tượng color

SolidBrush br = new SolidBrush(color); tạo đối tượng solid brush

Font font = new Font("Arial", 48, FontStyle.Bold); //tạo đối tượng font

StringFormat fm = new StringFormat(); //tạo đối tượng string format

fm.LineAlignment = StringAlignment.Far; //căn lề qua bên phải

g.DrawString("meo",font,br,rc,fm); //vẽ chuỗi trong hình chữ nhật rc1 vừa tạo

}

//================================

void DrawText(Rectangle rc, Graphics g) //Hàm vẽ chữ

{

LinearGradientBrush lbr = new LinearGradientBrush(rc, Color.Black, Color.White, 45); //tạo đổi tượng cọ vẽ linear

g.FillRectangle(lbr, rc); //tô màu đối tượng hình chữ nhật rc2

LinearGradientBrush br2 = new LinearGradientBrush(rc, Color.Orange, Color.Yellow, LinearGradientMode.Horizontal); //tạo đối tượng cọ vẽ linear br2

Font font = new Font("Arial", 48, FontStyle.Bold|FontStyle.Italic); //tạo đối tượng font

StringFormat fm = new StringFormat(); //tạo đối tượng Format

fm.LineAlignment = StringAlignment.Center; //chỉnh lề của đối tượng ra giữa

fm.Alignment = StringAlignment.Center;

g.DrawString("HELLO", font, br2, rc, fm); Vẽ chuỗi có chữ HELLO trong hình chữ nhật rc2

}

//==============================================

void DrawPolygon(Rectangle rc, Graphics g) //hàm vẽ đa giác

{

Point[] arrP =

{

new Point(rc.Left,rc.Height/4),

new Point(rc.Left + rc.Width/2,rc.Top),

new Point(rc.Left+rc.Width,rc.Height/4),

new Point(rc.Left+rc.Width/2,rc.Height)

}; //tạo mảng chứa các vị trí của đỉnh hình đa giác

GraphicsPath path = new GraphicsPath(); //tạo đối tượng GraphicPath

path.AddPolygon(arrP); //thêm mảng các điểm của hình đa giác vào path

PathGradientBrush br = new PathGradientBrush(path); //tạo cọ vẽ PathGradientBrush

br.CenterColor = Color.White; //thiết lập màu ở trung tâm cho hình đa giác

br.SurroundColors = new Color[] { Color.Yellow, Color.Red, Color.Cyan, Color.Green }; //thiết lập màu bao quanh bên ngoài cho hình đa giác

g.FillPolygon(br,arrP); //Tô màu hình đa giác

}

}

}

**Giữa kì IT01**

public Form1()

{

InitializeComponent();

}

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

{

DialogResult result = MessageBox.Show("Thông báo", "Đóng ứng dụng", MessageBoxButtons.YesNo,

MessageBoxIcon.Warning);

if (result == DialogResult.Yes)

{

Application.Exit();

}

}

private void Form1\_FormClosed(object sender, FormClosedEventArgs e)

{

DialogResult result = MessageBox.Show("Thông báo", "Đóng ứng dụng", MessageBoxButtons.YesNo,

MessageBoxIcon.Warning);

if (result == DialogResult.Yes)

{

Application.Exit();

}

}

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

{

XuLyChuoi xlchuoi = new XuLyChuoi();

xlchuoi.MdiParent = this;

xlchuoi.Show();

}

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

{

Ve ve = new Ve();

ve.MdiParent = this;

ve.Show();

}

}

Xử Lý Chuỗi

public partial class XuLyChuoi : Form

{

public XuLyChuoi()

{

InitializeComponent();

}

private void XuLyChuoi\_Load(object sender, EventArgs e)

{

txtNhapchuoi.Focus();

}

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

{

if (rdDau.Checked)

{

string chuoi = txtNhapchuoi.Text;

listChuoi.Items.Insert(0, chuoi);

}

else

{

string chuoi = txtNhapchuoi.Text;

listChuoi.Items.Add(chuoi);

}

}

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

{

for (int i = listChuoi.SelectedItems.Count - 1; i >= 0; i--)

listChuoi.Items.Remove(listChuoi.SelectedItems[i]);

}

Label lbHienThi = new Label();

private void listChuoi\_SelectedIndexChanged(object sender, EventArgs e)

{

// Kiểm tra xem có item nào được chọn hay không

if (listChuoi.SelectedIndex != -1)

{

// Lấy item được chọn

string selectedItem = listChuoi.SelectedItem.ToString();

lbHienThi.Text = selectedItem;

lbHienThi.AutoSize = true;

// Xóa nội dung Panel hiện tại

panelHienthi.Controls.Clear();

// Thêm Label vào Panel

panelHienthi.Controls.Add(lbHienThi);

}

}

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

{

// Tạo hộp thoại FontDialog

FontDialog font = new FontDialog();

// Hiển thị hộp thoại và kiểm tra kết quả

if (font.ShowDialog() == DialogResult.OK)

{

// Thay đổi font của Label

lbHienThi.Font = font.Font;

}

}

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

{

// Tạo hộp thoại ColorDialog

ColorDialog color = new ColorDialog();

// Hiển thị hộp thoại và kiểm tra kết quả

if (color.ShowDialog() == DialogResult.OK)

{

// Thay đổi màu của Label

lbHienThi.ForeColor = color.Color;

}

}

private string NormalizeString(string text)

{

// Loại bỏ khoảng trắng thừa đầu và cuối chuỗi

string trimedString = text.Trim();

// Loại bỏ khoảng trắng thừa giữa các từ

string[] words = trimedString.Split(new char[] { ' ' },

StringSplitOptions.RemoveEmptyEntries);

string normalizeString = string.Join(" ", words);

// Chuyển chữ cái đầu mỗi từ thành chữ hoa

normalizeString = System.Globalization.CultureInfo.CurrentCulture.TextInfo.

ToTitleCase(normalizeString.ToLower());

return normalizeString;

}

public static string NormalizeString1(string input)

{

if (string.IsNullOrWhiteSpace(input))

{

return string.Empty;

}

// Loại bỏ khoảng trắng thừa

string trimmed = input.Trim();

// Loại bỏ khoảng trắng thừa giữa các từ

string normalizedSpaces = Regex.Replace(trimmed, @"\s+", " ");

// Viết hoa chữ cái đầu của mỗi từ

TextInfo textInfo = CultureInfo.CurrentCulture.TextInfo;

string capitalized = textInfo.ToTitleCase(normalizedSpaces.ToLower());

return capitalized;

}

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

{

// Lấy chuỗi từ Lable

string inputString = lbHienThi.Text;

// Chuẩn hóa chuỗi

string normalizeString = NormalizeString(inputString);

lbHienThi.Text = normalizeString;

}

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

{

// Lấy chuỗi từ Lable

string inputString = lbHienThi.Text;

//Đảo chuỗi

string reverse = ReserveWords(inputString);

//Hiển thị kết quả

lbHienThi.Text = reverse;

}

//Đảo từ

private string ReserveWords(string words)

{

// Tách chuỗi thành các từ

string[] word = words.Split(' ');

// Đảo ngược mảng các từ

Array.Reverse(word);

// Nối các từ lại thành chuỗi

return string.Join(" ", word);

}

//Đảo kí tự

private string DaoChuoi(string inputString)

{

char[] charArray = inputString.ToCharArray();

Array.Reverse(charArray);

return new string(charArray);

}

}

public partial class Ve : Form

{

public Ve()

{

InitializeComponent();

}

private void Ve\_Paint(object sender, PaintEventArgs e)

{

Rectangle rc1 = new Rectangle(0, 0, ClientRectangle.Width / 2,

ClientRectangle.Height / 2);

Rectangle rc2 = new Rectangle(ClientRectangle.Width / 2, 0,

ClientRectangle.Width / 2, ClientRectangle.Height / 2);

Rectangle rc3 = new Rectangle(0, ClientRectangle.Height / 2,

ClientRectangle.Width, ClientRectangle.Height / 2);

DrawImagesLeft(rc1, e.Graphics);

DrawImagesRight(rc2, e.Graphics);

DrawText(rc3, e.Graphics);

Pen pen = new Pen(Color.Red, 3);

e.Graphics.DrawRectangles(pen, new Rectangle[] { rc1, rc2, rc3 });

}

void DrawImagesLeft(Rectangle rc, Graphics g)

{

Image img = Image.FromFile("vet.jpg");

g.DrawImage(img, rc);

}

void DrawImagesRight(Rectangle rc, Graphics g)

{

Image img = Image.FromFile("vet.jpg");

img.RotateFlip(RotateFlipType.RotateNoneFlipX);

g.DrawImage(img, rc);

}

void DrawText(Rectangle rc, Graphics g)

{

HatchBrush hbr = new HatchBrush(HatchStyle.Cross,Color.Orange,Color.Yellow);

g.FillRectangle(hbr, rc);

Font font = new Font("Arial", 64, FontStyle.Bold);

string text = "2251050056";

float x = rc.Left +(rc.Width-g.MeasureString(text,font).Width) / 2;

float y = rc.Top +(rc.Height-g.MeasureString(text, font).Height)/2;

Color shadowColor = Color.FromArgb(200, Color.Gray);

using (SolidBrush shadowBrush = new SolidBrush(shadowColor))

{

g.DrawString(text, font, shadowBrush, new RectangleF(x + 5, y + 5, rc.Width, rc.Height));

}

g.DrawString(text, font, Brushes.Blue, x, y);

}

private void Ve\_SizeChanged(object sender, EventArgs e)

{

Invalidate();

}

}

**Đề 1(Trò chơi Quay Số Xúc Xắc-Vẽ Ảnh Ngược, MSSV)**

public partial class Login : Form

{

public Login()

{

InitializeComponent();

}

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

{

Form1 form = new Form1();

form.MdiParent = this;

form.Show();

}

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

{

Ve ve = new Ve();

ve.MdiParent = this;

ve.Show();

}

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

{

// Hiển thị hộp thoại cảnh báo

DialogResult result = MessageBox.Show("Close Application?", "Warning",

MessageBoxButtons.YesNo, MessageBoxIcon.Warning);

// Kiểm tra lựa chọn của người dùng

if (result == DialogResult.Yes)

{

// Hủy sự kiện đóng form

Application.Exit();

}

}

}

public partial class Form1 : Form

{

string pathImg;

Random rand = new Random();

int Count, nNum, mNum, lNum;

bool bClose = false;

public Form1()

{

InitializeComponent();

}

private void Form1\_Load(object sender, EventArgs e)

{

Init();

lbChu.Text = " TRÒ CHƠI QUAY SỐ ";

timer1.Start();

}

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

{

Play();

}

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

{

Init();

}

private void Form1\_FormClosing(object sender, FormClosingEventArgs e)

{

if (!bClose)

{

e.Cancel = true;

}

}

private void timer1\_Tick(object sender, EventArgs e)

{

lbChu.Text = lbChu.Text[lbChu.Text.Length - 1] +

lbChu.Text.Substring(0, lbChu.Text.Length - 1);

}

private void Init()

{

Count = nNum = mNum = lNum = 0;

pathImg = Application.StartupPath + @"\hinhxucxac\";

picBox1.Image = Image.FromFile(pathImg + "6.jpg");

picBox2.Image = Image.FromFile(pathImg + "6.jpg");

picBox3.Image = Image.FromFile(pathImg + "6.jpg");

ListResult.Items.Clear();

}

void Play()

{

Count++;

int n = rand.Next(1, 7);

picBox1.Image = Image.FromFile(pathImg + n.ToString() + ".jpg");

nNum = 1;

int m = rand.Next(1, 7);

picBox2.Image = Image.FromFile(pathImg + m.ToString() + ".jpg");

mNum = 1;

int l = rand.Next(1, 7);

picBox3.Image = Image.FromFile(pathImg + l.ToString() + ".jpg");

lNum = 1;

//trên giao diện

if (n == m && n == l && m == l)

{

nNum += lNum + mNum;

ListResult.Items.Add(String.Format(" Lần {0}: Số {1}: {2} lần. ", Count, n,

nNum));

}

else if (n == m)

{

nNum += mNum;

ListResult.Items.Add(String.Format(" Lần {0}: Số {1}: {2} lần. " +

"Số {3}: {4} lần", Count, n, nNum, l, lNum));

}

else if (n == l)

{

nNum += lNum;

ListResult.Items.Add(String.Format(" Lần {0}: Số {1}: {2} lần. " +

"Số {3}: {4} lần", Count, n, nNum, m, mNum));

}

else if (l == m)

{

mNum += lNum;

ListResult.Items.Add(String.Format(" Lần {0}: Số {1}: {2} lần. " +

"Số {3}: {4} lần", Count, n, nNum, m, mNum));

}

else

{

ListResult.Items.Add(String.Format(" Lần {0}: Số {1}: {2} lần. " +

"Số {3}: {4} lần. Số {5}: {6} lần", Count, n, nNum, m, mNum, l, lNum));

}

}

}

public partial class Ve : Form

{

public Ve()

{

InitializeComponent();

}

private void Ve\_Paint(object sender, PaintEventArgs e)

{

Rectangle rc1 = new Rectangle(0, 0, ClientRectangle.Width / 2,

ClientRectangle.Height / 2);

Rectangle rc2 = new Rectangle(ClientRectangle.Width / 2, 0,

ClientRectangle.Width / 2, ClientRectangle.Height / 2);

Rectangle rc3 = new Rectangle(0, ClientRectangle.Height / 2,

ClientRectangle.Width, ClientRectangle.Height / 2);

DrawImagesLeft(rc1, e.Graphics);

DrawImagesRight(rc2, e.Graphics);

DrawText(rc3, e.Graphics);

Pen pen = new Pen(Color.Red, 3);

e.Graphics.DrawRectangles(pen, new Rectangle[] { rc1, rc2, rc3 });

}

void DrawImagesLeft(Rectangle rc, Graphics g)

{

Image img = Image.FromFile("vet.jpg");

g.DrawImage(img, rc);

}

void DrawImagesRight(Rectangle rc, Graphics g)

{

Image img = Image.FromFile("vet.jpg");

img.RotateFlip(RotateFlipType.RotateNoneFlipX);

g.DrawImage(img, rc);

}

// Vẽ MSSV và có Bóng Mờ

void DrawText(Rectangle rc, Graphics g)

{

HatchBrush hbr = new HatchBrush(HatchStyle.Cross,Color.Orange,Color.Yellow);

g.FillRectangle(hbr, rc);

Font font = new Font("Arial", 64, FontStyle.Bold);

string text = "2251050056";

float x = rc.Left +(rc.Width-g.MeasureString(text,font).Width) / 2;

float y = rc.Top +(rc.Height-g.MeasureString(text, font).Height)/2;

Color shadowColor = Color.FromArgb(200, Color.Gray);

using (SolidBrush shadowBrush = new SolidBrush(shadowColor))

{

g.DrawString(text, font, shadowBrush, new RectangleF(x + 5, y + 5, rc.Width, rc.Height));

}

g.DrawString(text, font, Brushes.Blue, x, y);

}

private void Ve\_SizeChanged(object sender, EventArgs e)

{

Invalidate();

}

}

**Đề 2(Vẽ HCN-Hình Tròn)**

**// Form Main**

public partial class FrmMain : Form

{

public FrmMain()

{

InitializeComponent();

}

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

{

FrmQuaySo frm = new FrmQuaySo();

frm.Show();

frm.MdiParent = this;

}

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

{

FrmVe frm = new FrmVe();

frm.Show();

frm.MdiParent = this;

}

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

{

Application.Exit();

}

private void FrmMain\_FormClosing(object sender, FormClosingEventArgs e)

{

if (MessageBox.Show("Đóng chương trình?","Cảnh báo!",MessageBoxButtons.YesNo,MessageBoxIcon.Warning)

==DialogResult.No)

{

e.Cancel = true;

}

}

}

**//Form Vẽ**

public partial class FrmVe : Form

{

public FrmVe()

{

InitializeComponent();

}

Color penColor=Color.Black;

Color backColor=Color.Black;

int pWidth=1;

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

{

if (txtX.Text!="" && txtY.Text!="" && txtHeight.Text!="" && txtWidth.Text!="")

{

ListViewItem item = new ListViewItem(txtX.Text);

item.SubItems.Add(txtY.Text);

item.SubItems.Add(txtWidth.Text);

item.SubItems.Add(txtHeight.Text);

listView1.Items.Add(item);

txtX.Text = "";

txtY.Text = "";

txtWidth.Text = "";

txtHeight.Text = "";

txtX.Focus();

}

}

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

{

foreach (ListViewItem item in listView1.SelectedItems)

{

listView1.Items.Remove(item);

}

}

private void lbMauValue\_DoubleClick(object sender, EventArgs e)

{

ColorDialog color = new ColorDialog();

if (color.ShowDialog() == DialogResult.OK)

{

lbMauValue.BackColor = color.Color;

backColor = color.Color;

}

}

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

{

ColorDialog color = new ColorDialog();

if (color.ShowDialog() == DialogResult.OK)

{

lbMauVienValue.BackColor = color.Color;

penColor = color.Color;

}

}

private void numericUpDown1\_ValueChanged(object sender, EventArgs e)

{

pWidth = (int)numericUpDown1.Value;

}

private void listView1\_MouseClick(object sender, MouseEventArgs e)

{

Graphics g = panel1.CreateGraphics();

int x = int.Parse(listView1.SelectedItems[0].Text);

int y = int.Parse(listView1.SelectedItems[0].SubItems[1].Text);

int w = int.Parse(listView1.SelectedItems[0].SubItems[2].Text);

int h = int.Parse(listView1.SelectedItems[0].SubItems[3].Text);

Pen pen = new Pen(penColor, pWidth);

SolidBrush slbr = new SolidBrush(backColor);

Rectangle rec = new Rectangle(x, y, w, h);

g.DrawRectangle(pen, rec);

g.FillEllipse(slbr, rec);

}

private void listView1\_ItemSelectionChanged(object sender, ListViewItemSelectionChangedEventArgs e)

{

panel1.Invalidate();

}

}