**Tên: Nguyễn Anh Khôi**

**MSSV: 1150080141**

**LAB 2:**

**A. Mục tiêu**

− Hướng dẫn sinh viên xây dựng chương trình Windows Form đơn giản. − Hướng dẫn làm quen môi trường lập trình C# với phần mềm Microsoft Visual Studio 2022.

− Hướng dẫn sinh viên tập làm quen mới soạn thảo, chỉnh sửa mã nguồn, biên dịch, gỡ lỗi (debug), thực thi chương trình.

− Làm quen với các control cơ bản trên Winform như:

+ Label: hiển thị thông tin.

+ TextBox: nhập thông tin dữ liệu.

+ Button: cho phép người dùng click để thực hiện chức năng nào đó. + CheckBox: cho phép người dùng chọn một hoặc nhiều tùy chọn. + RadioButton: cho phép người dùng chọn một tùy chọn.

+ MessageBox: hiển thị thông tin đến người dùng theo dạng hộp thoại. + NumericUpDown: Hộp chọn tăng giảm giá trị số.

+ Combobox: Hộp chọn 1 giá trị trong danh sách giá trị.

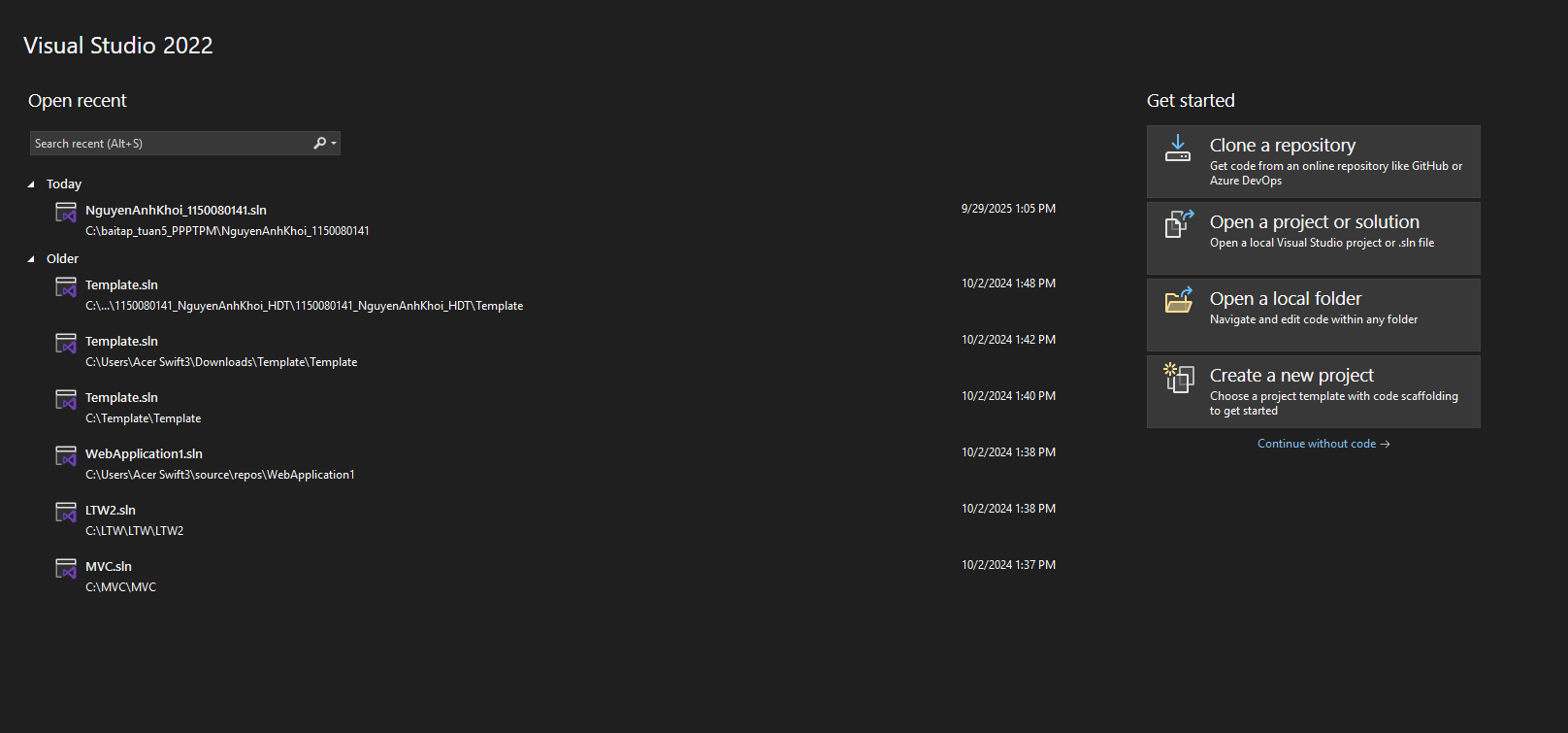
+ ListBox: Danh sách các mục chọn, cho phép chọn 1 hoặc nhiều mục. + GroupBox: Nhóm các đối tượng về cùng nhóm.

− Làm quen khai báo và xử lý các sự kiện với các control khác nhau. − Làm quen với cách đặt tên biến lập trình của các control.

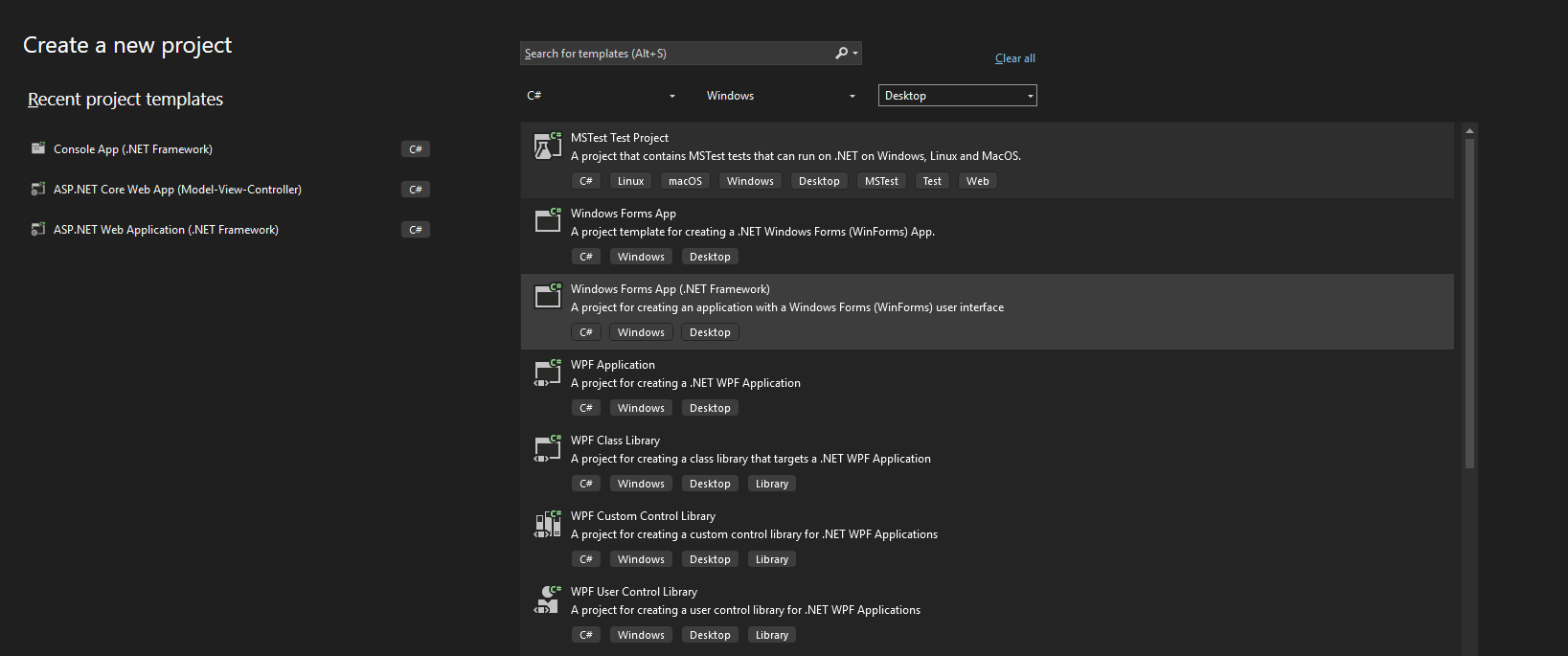
**B. Nội dung**

**1. Tạo Project Windows Forms Application trên Microsoft visualstudio 2022**

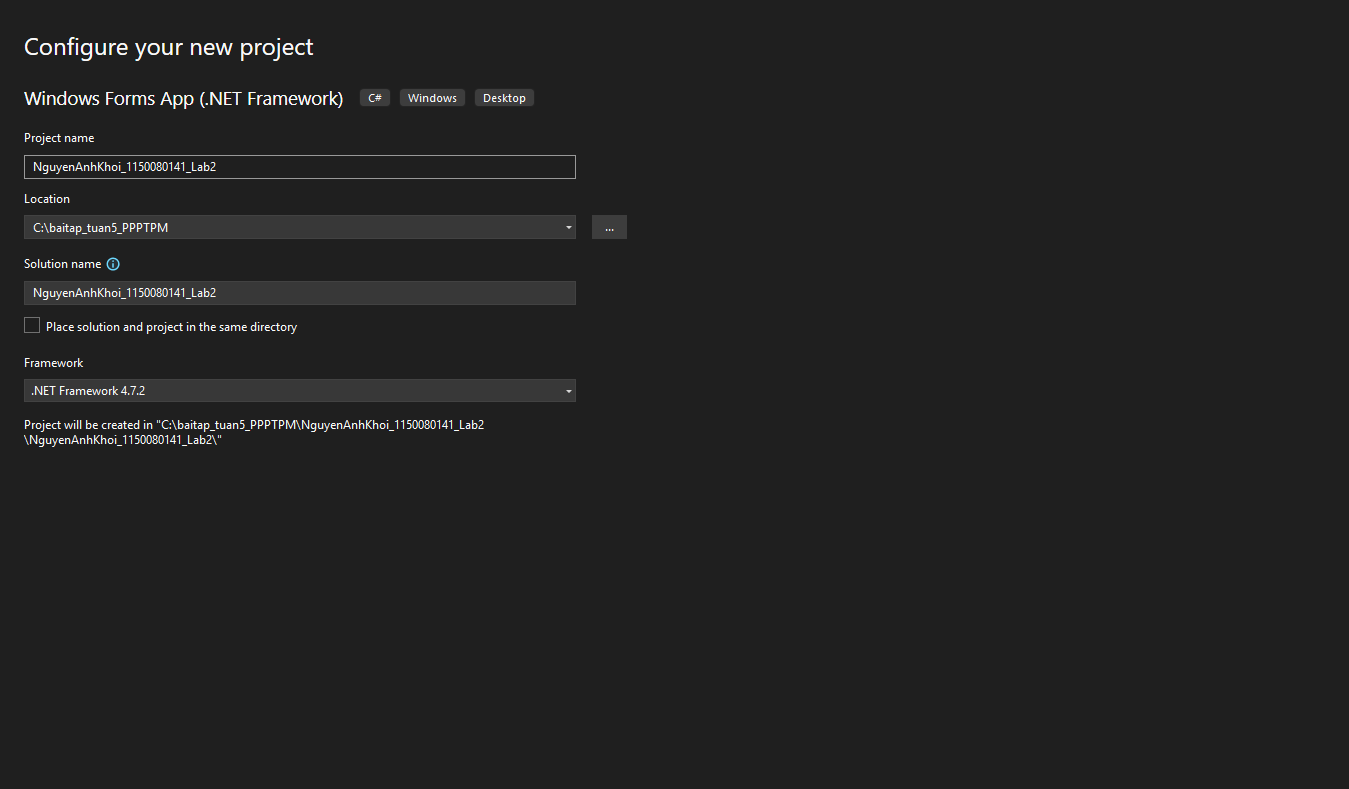
**Bước 1:** Khởi động chương trình visual studio 2022. Hình ảnh dưới đây là màn hình khởi động của ứng dụng:

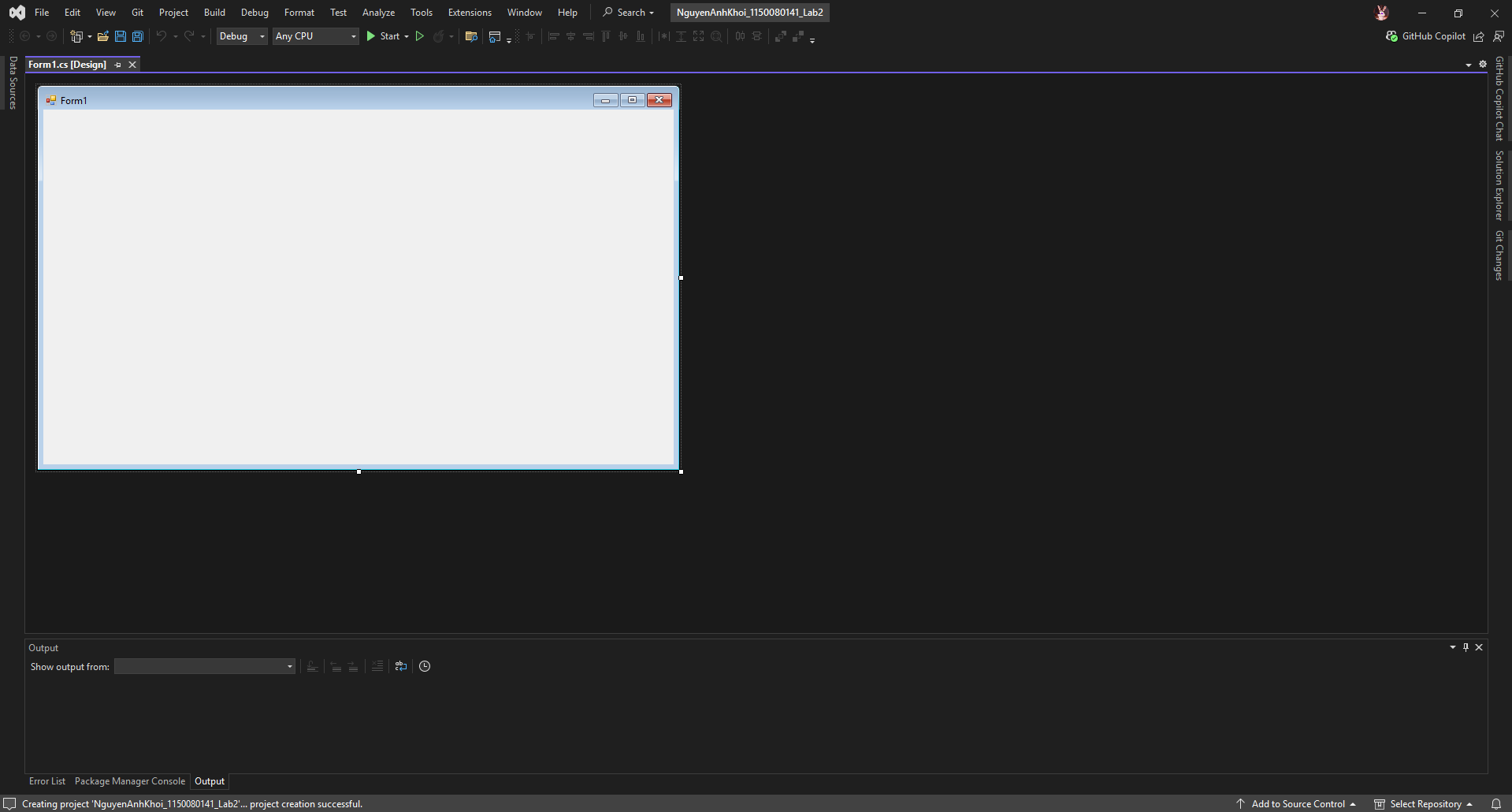


**Bước 2:** Chọn Create a new project trên màn hình khởi động. Trong màn hình tiếp theo, ở mục All languages → chọn C# trong danh sách ngôn ngữ lập trình. Mục All platforms → chọn Windows. Ở mục All project types → chọn Desktop.



**Bước 3:** Trong cửa sổ tiếp theo, bạn điền đầy đủ các thông tin về Project.

Đây là kết quả!



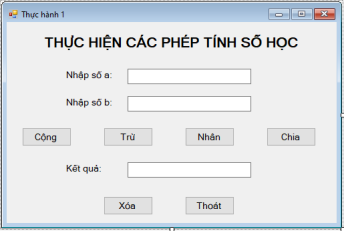
Màn hình làm việc cho ứng dụng Winform bao gồm các phần cơ bản sau:

Số 1: Toolbox → chứa các control cho phép kéo thả vào form.

Số 2: màn hình thiết kế → nơi thiết kế trực quan, có thể chuyển đổi qua lại giữa code editor và giao diện.

Số 3: Solution Exlorer → lập trình viên quản lý các thành phần trong dự án. Số 4: Property → cho phép tùy chỉnh lại các thành phần của control và form. **2. Thực hành**

**Thực hành 1:** Thiết kế form như sau:



Form này sẽ cho phép người dùng nhập các giá trị là số thực vào 2 biến a và b. Tùy vào các chức năng của các phép toán (cộng, trừ, nhân, chia) mà hiển thị ở phần kết quả.

Đặt tên biến cho một số control như sau:

− TextBox cho phép nhập a: txtA

− TextBox cho phép nhập b: txtB

− TextBox cho phép hiển thị kết quả phép tính: txtKetQua

− Button phép toán cộng: btnCong

− Button phép toán trừ: btnTru

− Button phép toán nhân: btnNhan

− Button phép toán chia: btnChia

− Button phép toán xóa dữ liệu: btnXoa

− Button phép toán thoát chương trình: btnThoat

Tham khảo hướng dẫn lập trình cho Phép toán Cộng:

using System;

using System.Windows.Forms;

namespace Lab2\_TH1

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

// Hàm chuyển đổi số

private double GetValue(TextBox txt)

{

double value = 0;

if (!double.TryParse(txt.Text, out value))

{

MessageBox.Show("Vui lòng nhập số hợp lệ!");

}

return value;

}

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

{

double a = GetValue(txtA);

double b = GetValue(txtB);

txtKetQua.Text = (a + b).ToString();

}

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

{

double a = GetValue(txtA);

double b = GetValue(txtB);

txtKetQua.Text = (a - b).ToString();

}

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

{

double a = GetValue(txtA);

double b = GetValue(txtB);

txtKetQua.Text = (a \* b).ToString();

}

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

{

double a = GetValue(txtA);

double b = GetValue(txtB);

if (b == 0)

{

MessageBox.Show("Không thể chia cho 0!");

return;

}

txtKetQua.Text = (a / b).ToString();

}

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

{

txtA.Clear();

txtB.Clear();

txtKetQua.Clear();

txtA.Focus();

}

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

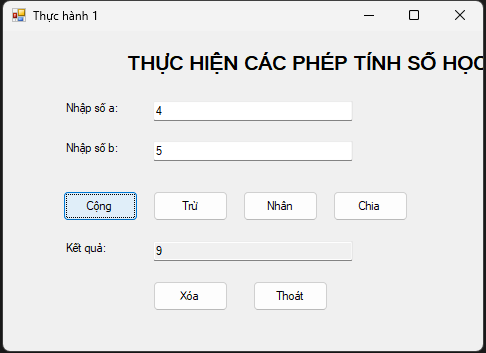
{

Application.Exit();

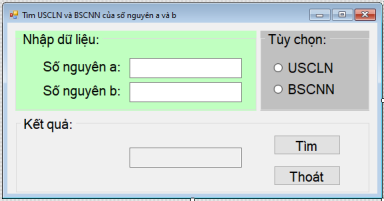
}

}

}



**Thực hành 2:** Thêm vào project là “ApDung1”, thực hiện tìm ước số chung lớn nhất và bội số chung nhỏ nhất của 2 số nguyên a và b:

**Yêu cầu:**

− Viết 2 hàm để tính ước số chung lớn nhất và bội số chung nhỏ nhất của 2 số a và b. Khi nhấn vào nút Tìm, nếu người dùng chọn tùy chọn tìm USCLN thì hiển thị ước số chung lớn nhất, ngược lại hiển thị bội số chung nhỏ nhất.

− Nút thoát để thoát khỏi chương trình.

using System;

using System.Numerics;

using System.Windows.Forms;

namespace ApDung1

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

radUSCLN.Checked = true;

}

private long USCLN(long a, long b)

{

a = Math.Abs(a); b = Math.Abs(b);

if (a == 0) return b;

if (b == 0) return a;

while (b != 0)

{

long r = a % b;

a = b;

b = r;

}

return a;

}

private long BSCNN(long a, long b)

{

a = Math.Abs(a); b = Math.Abs(b);

if (a == 0 || b == 0) return 0;

long g = USCLN(a, b);

checked { return (a / g) \* b; }

}

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

{

if (!long.TryParse(txtA.Text.Trim(), out long a) ||

!long.TryParse(txtB.Text.Trim(), out long b))

{

MessageBox.Show("Vui lòng nhập hai số nguyên hợp lệ!", "Lỗi nhập liệu",

MessageBoxButtons.OK, MessageBoxIcon.Warning);

return;

}

try

{

if (radUSCLN.Checked)

txtKetQua.Text = $"USCLN({a}, {b}) = {USCLN(a, b)}";

else

txtKetQua.Text = $"BSCNN({a}, {b}) = {BSCNN(a, b)}";

}

catch (OverflowException)

{

MessageBox.Show("Kết quả quá lớn, xảy ra tràn số!", "Lỗi",

MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

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

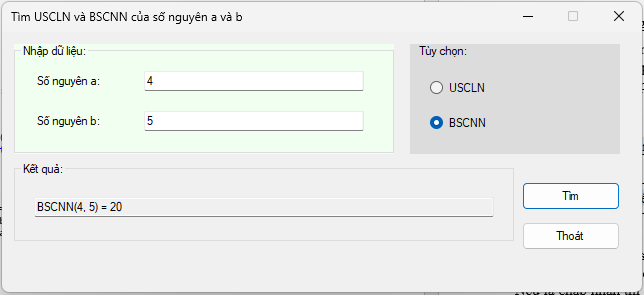
{

Application.Exit();

}

}

}



**Thực hành 3:** Xây dựng ứng dụng mô phỏng thiết bị Security Panel theo yêu cầu:

Một phòng Lab muốn lắp thiết bị bảo vệ đặt được ở bên ngoài cửa ra vào có tên là Security. Chỉ cho phép những cá nhân nào nhập đúng password đã được quy định mới được vào phòng. Bảng dưới là password được cấp cho những nhóm thành viên:

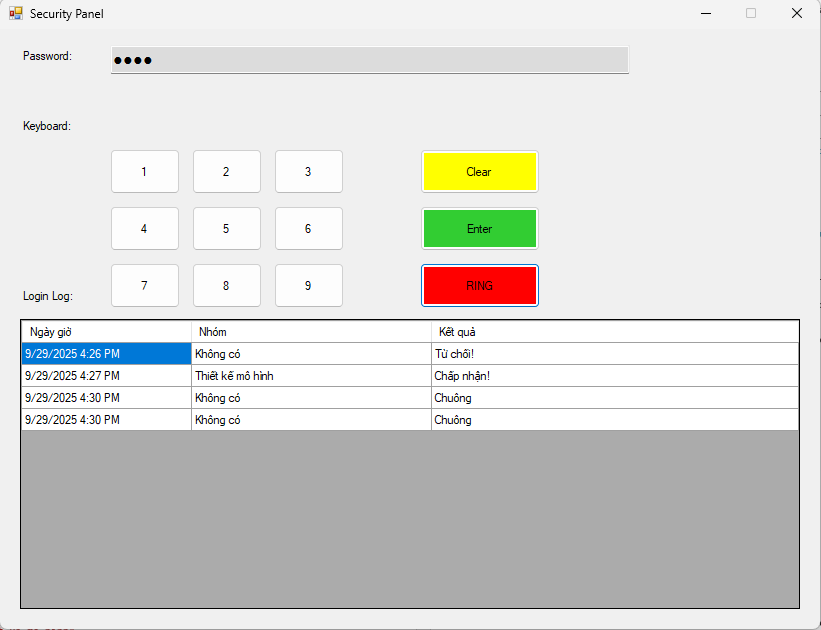
**Nhóm thành viên Password**

Phát triển công nghệ 1496 hoặc 2673

|  |  |
| --- | --- |
| Nghiên cứu viên | 7462 |
| Thiết kế mô hình | 8884 hoặc 3842 hoặc 3383 |

Mỗi một lần nhập password sẽ có 2 trạng thái: chấp nhận hoặc từ chối. Tất cả các thông tin truy cập sẽ được hiển thị trong khung thông tin phía dưới bàn phím số. Nếu là chấp nhận thì ghi lại và hiển thị thông về về ngày, thời gian, tên nhóm và chấp nhận. Nếu là từ chối thì ghi lại và hiển thị thông về về ngày, thời gian, từ chối.

Giao diện ứng dụng:



using System;

using System.Collections.Generic;

using System.Linq;

using System.Media;

using System.Windows.Forms;

using static System.Windows.Forms.VisualStyles.VisualStyleElement.Window;

namespace SecurityPanel

{

public partial class Form1 : Form

{

// Map password -> group

private readonly Dictionary<string, string> \_passToGroup =

new Dictionary<string, string>

{

["1496"] = "Phát triển công nghệ",

["2673"] = "Phát triển công nghệ",

["7462"] = "Nghiên cứu viên",

["8884"] = "Thiết kế mô hình",

["3842"] = "Thiết kế mô hình",

["3383"] = "Thiết kế mô hình"

};

public Form1()

{

InitializeComponent();

txtPassword.MaxLength = 4;

txtPassword.PasswordChar = '●';

// gán cùng 1 handler cho tất cả nút số

foreach (var btn in new[] { btn1, btn2, btn3, btn4, btn5, btn6, btn7, btn8, btn9 })

btn.Click += Digit\_Click;

btnClear.Click += (s, e) => txtPassword.Clear();

btnRing.Click += (s, e) => { SystemSounds.Beep.Play(); Log("Không có", "Chuông"); };

btnEnter.Click += BtnEnter\_Click;

// chuẩn bị DataGridView

dgvLog.AutoGenerateColumns = false;

dgvLog.Columns.Clear();

dgvLog.Columns.Add(new DataGridViewTextBoxColumn { HeaderText = "Ngày giờ", DataPropertyName = "Time", Width = 170 });

dgvLog.Columns.Add(new DataGridViewTextBoxColumn { HeaderText = "Nhóm", DataPropertyName = "Group", Width = 240 });

dgvLog.Columns.Add(new DataGridViewTextBoxColumn { HeaderText = "Kết quả", DataPropertyName = "Result", AutoSizeMode = DataGridViewAutoSizeColumnMode.Fill });

dgvLog.DataSource = \_items;

}

private class LogItem { public string Time { get; set; } public string Group { get; set; } public string Result { get; set; } }

private readonly BindingSource \_items = new BindingSource();

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

{

if (txtPassword.TextLength >= txtPassword.MaxLength) return;

var btn = (Button)sender;

txtPassword.Text += btn.Tag?.ToString() ?? btn.Text;

}

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

{

var pwd = txtPassword.Text.Trim();

if (pwd.Length == 0) return;

if (\_passToGroup.TryGetValue(pwd, out var group))

{

Log(group, "Chấp nhận!");

}

else

{

Log("Không có", "Từ chối!");

}

txtPassword.Clear();

}

private void Log(string group, string result)

{

\_items.Add(new LogItem

{

Time = DateTime.Now.ToString("g"),

Group = group,

Result = result

});

dgvLog.FirstDisplayedScrollingRowIndex = dgvLog.RowCount - 1;

}

}

}

**Thực hành 4:** Tính tiền công dịch vụ tại phòng khám nha khoa.

Giả sử, tại một phòng khám nha khoa cần tính tiền các loại dịch vụ khác nhau. Một biểu mẫu thanh toán gồm:

− Tên khách hàng.

− Các loại dịch vụ như: lấy cao răng (50.000đ), tẩy trắng răng (100.000đ), hàn răng (100.000đ), bẻ răng (10.000đ), bọc răng (1.000.000đ).

− Hàn răng, bẻ răng và bọc răng được tính theo chiếc răng. Giá tiền trên được tính theo 1 chiếc răng.

− Biểu mẫu tính tiền không được để trống tên khách hàng. Có cảnh báo nếu trống.

Tham khảo giao diện ứng dụng sau:



using System;

using System.ComponentModel;

using System.Globalization;

using System.Windows.Forms;

namespace NhaKhoa

{

public partial class Form1 : Form

{

const int GIA\_LAY\_CAO = 50\_000;

const int GIA\_TAY\_TRANG = 100\_000;

const int GIA\_HAN = 100\_000;

const int GIA\_BE = 10\_000;

const int GIA\_BOC = 1\_000\_000;

readonly CultureInfo viVN = new CultureInfo("vi-VN");

public Form1()

{

InitializeComponent();

foreach (var n in new[] { nupHanRang, nupBeRang, nupBocRang })

{

n.Minimum = 1;

n.Maximum = 100;

n.Value = 1;

n.Enabled = false;

}

chkHanRang.CheckedChanged += (s, e) => nupHanRang.Enabled = chkHanRang.Checked;

chkBeRang.CheckedChanged += (s, e) => nupBeRang.Enabled = chkBeRang.Checked;

chkBocRang.CheckedChanged += (s, e) => nupBocRang.Enabled = chkBocRang.Checked;

txtTenKH.Validating += txtTenKH\_Validating;

btnTinhTien.Click += btnTinhTien\_Click;

btnThoat.Click += (s, e) => Close();

}

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

{

if (!ValidateChildren(ValidationConstraints.Enabled)) return;

double thanhTien = 0;

if (chkLayCaoRang.Checked) thanhTien += GIA\_LAY\_CAO;

if (chkTayTrangRang.Checked) thanhTien += GIA\_TAY\_TRANG;

if (chkHanRang.Checked) thanhTien += (int)nupHanRang.Value \* GIA\_HAN;

if (chkBeRang.Checked) thanhTien += (int)nupBeRang.Value \* GIA\_BE;

if (chkBocRang.Checked) thanhTien += (int)nupBocRang.Value \* GIA\_BOC;

txtThanhTien.Text = thanhTien.ToString("#,0", viVN) + " đ";

}

private void txtTenKH\_Validating(object sender, CancelEventArgs e)

{

if (string.IsNullOrWhiteSpace(txtTenKH.Text))

{

e.Cancel = true;

err.SetError(txtTenKH, "Tên khách hàng không được để trống!");

}

else

{

e.Cancel = false;

err.SetError(txtTenKH, null);

}

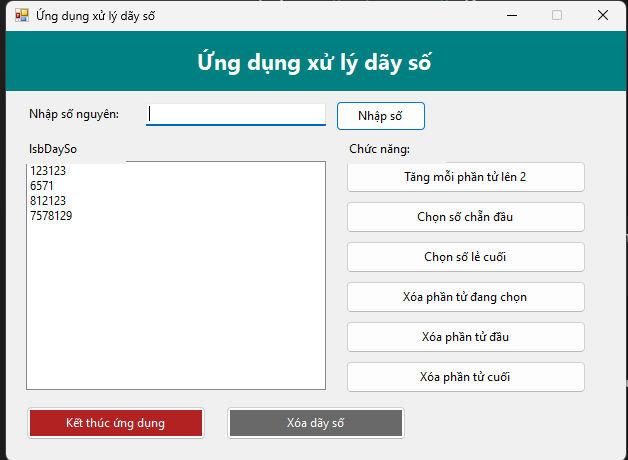
}

}

}

**Áp dụng 3:** Thiết kế và lập trình form Đăng nhập của một ứng dụng. Thông tin cần đăng nhập gồm username và password. Yêu cầu người dùng không được để trống trường thông tin nào. Nếu để trống trường thông tin nào thì cảnh báo người dùng để bổ sung.

**Thực hành 5:** Thiết kế form frmXuLySo với control ListBox như sau:



**Yêu cầu:**

− Chỉ cho nhập số vào textbox Nhập số nguyên

− Khi nhập dữ liệu vào textbox Nhập số nguyên và nhấn Enter (hoặc click vào nút Nhập số) thì số mới nhập này được thêm vào Listbox đồng thời dữ liệu trong textbox bị xóa và focus được chuyển về textbox

− Khi nhấn vào các phím chức năng, yêu cầu sẽ được thực hiện trên listbox.

using System;

using System.Windows.Forms;

namespace XuLyDaySoApp

{

public partial class frmXuLySo : Form

{

public frmXuLySo()

{

InitializeComponent();

Load += (s, e) => lsbDaySo.Items.Clear();

// chỉ cho nhập số nguyên (hỗ trợ dấu - ở đầu & Backspace)

txtInput.KeyPress += (s, e) =>

{

if (char.IsControl(e.KeyChar)) return;

if (char.IsDigit(e.KeyChar)) return;

if (e.KeyChar == '-' && txtInput.SelectionStart == 0 && !txtInput.Text.Contains("-")) return;

e.Handled = true;

};

// Enter để thêm

txtInput.KeyDown += (s, e) => { if (e.KeyCode == Keys.Enter) { btnInput.PerformClick(); e.SuppressKeyPress = true; } };

btnInput.Click += btnInput\_Click;

btnTang2.Click += btnTang2\_Click;

btnChanDau.Click += btnChanDau\_Click;

btnLeCuoi.Click += btnLeCuoi\_Click;

btnXoaChon.Click += btnXoaChon\_Click;

btnXoaDau.Click += btnXoaDau\_Click;

btnXoaCuoi.Click += btnXoaCuoi\_Click;

btnXoaDaySo.Click += (s, e) => lsbDaySo.Items.Clear();

btnClose.Click += (s, e) =>

{

var result = MessageBox.Show("Bạn có thực sự muốn thoát?", "Hỏi thoát!",

MessageBoxButtons.YesNo, MessageBoxIcon.Question);

if (result == DialogResult.Yes) Close();

};

AcceptButton = btnInput;

}

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

{

if (int.TryParse(txtInput.Text.Trim(), out int val))

{

lsbDaySo.Items.Add(val);

txtInput.Clear();

txtInput.Focus();

}

else

{

MessageBox.Show("Không phải số nguyên! Hãy nhập lại!", "Thông báo",

MessageBoxButtons.OK, MessageBoxIcon.Warning);

txtInput.Clear(); txtInput.Focus();

}

}

private void btnTang2\_Click(object sender, EventArgs e)

{

if (lsbDaySo.Items.Count == 0)

{

MessageBox.Show("Dãy số đang trống. Vui lòng nhập dữ liệu!",

"Thông báo!", MessageBoxButtons.OK, MessageBoxIcon.Error);

txtInput.Focus(); return;

}

for (int i = 0; i < lsbDaySo.Items.Count; i++)

lsbDaySo.Items[i] = (int)lsbDaySo.Items[i] + 2;

}

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

{

if (lsbDaySo.Items.Count == 0) { EmptyWarn(); return; }

for (int i = 0; i < lsbDaySo.Items.Count; i++)

if ((int)lsbDaySo.Items[i] % 2 == 0) { lsbDaySo.SelectedIndex = i; break; }

}

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

{

if (lsbDaySo.Items.Count == 0) { EmptyWarn(); return; }

for (int i = lsbDaySo.Items.Count - 1; i >= 0; i--)

if ((int)lsbDaySo.Items[i] % 2 != 0) { lsbDaySo.SelectedIndex = i; break; }

}

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

{

if (lsbDaySo.Items.Count == 0) { EmptyWarn(); return; }

if (lsbDaySo.SelectedIndices.Count == 0)

{

MessageBox.Show("Bạn chưa chọn số cần xóa!", "Thông báo!",

MessageBoxButtons.OK, MessageBoxIcon.Error);

return;

}

while (lsbDaySo.SelectedIndex != -1)

lsbDaySo.Items.RemoveAt(lsbDaySo.SelectedIndex);

}

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

{

if (lsbDaySo.Items.Count == 0) { EmptyWarn(); return; }

lsbDaySo.Items.RemoveAt(0);

}

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

{

if (lsbDaySo.Items.Count == 0) { EmptyWarn(); return; }

lsbDaySo.Items.RemoveAt(lsbDaySo.Items.Count - 1);

}

private void EmptyWarn()

{

MessageBox.Show("Dãy số đang trống. Vui lòng nhập dữ liệu!",

"Thông báo!", MessageBoxButtons.OK, MessageBoxIcon.Error);

txtInput.Focus();

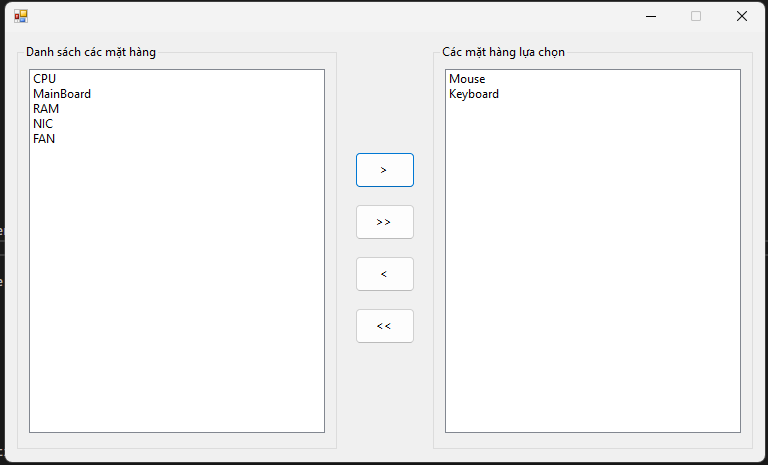
}

}

}

break;

**Thực hành 6:**

using System;

using System.Linq;

using System.Windows.Forms;

namespace ChuyenListBox

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

lstMatHang.Items.AddRange(new object[]

{ "CPU", "MainBoard", "RAM", "Keyboard", "Mouse", "NIC", "FAN" });

btnRight.Click += (s, e) => MoveSelected(lstMatHang, lstDaChon);

btnAllRight.Click += (s, e) => MoveAll(lstMatHang, lstDaChon);

btnLeft.Click += (s, e) => MoveSelected(lstDaChon, lstMatHang);

btnAllLeft.Click += (s, e) => MoveAll(lstDaChon, lstMatHang);

lstMatHang.DoubleClick += (s, e) => MoveSelected(lstMatHang, lstDaChon);

lstDaChon.DoubleClick += (s, e) => MoveSelected(lstDaChon, lstMatHang);

}

private void MoveSelected(ListBox src, ListBox dst)

{

if (src.SelectedItems.Count == 0) return;

var toMove = src.SelectedItems.Cast<object>().ToArray();

dst.Items.AddRange(toMove);

for (int i = src.SelectedIndices.Count - 1; i >= 0; i--)

src.Items.RemoveAt(src.SelectedIndices[i]);

}

private void MoveAll(ListBox src, ListBox dst)

{

if (src.Items.Count == 0) return;

dst.Items.AddRange(src.Items.Cast<object>().ToArray());

src.Items.Clear();

}

}

}