МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Криворізький національний університет

Кафедра моделювання та програмного забезпечення

Звіт

з лабораторної роботи №4

РОЗРОБКА ТА УПРАВЛІННЯ ІНТЕРАКТИВНИМ ІНТЕРФЕЙСОМ КОРИСТУВАЧА

Студент групи ІПЗ-21-2

Губарєв Р.В.

+380980190289

Викладачі

Козиков А. В.

Гриценко А. М.

Кривий Ріг

2022

1. **Короткі теоретичні відомості про візуальні об’єкти (компонентів), що використовуються.**

**MenuStrip -** Створює меню, що настроюється.

**DataGridView -** Елемент керування DataGridView надає таблицю, що настроюється, для відображення даних. Клас DataGridView дозволяє налаштовувати комірки, рядки, стовпці та межі.

**Button -** Запускає, зупиняє чи перериває процес.

**Panel -** Групують набір елементів керування на кадрі без міток з можливістю прокручування.

**CheckBox -** Відображає прапорець та мітку для тексту. Зазвичай використовується завдання параметрів.

**RadioButton -** Відображає кнопку, яку можна увімкнути або вимкнути.

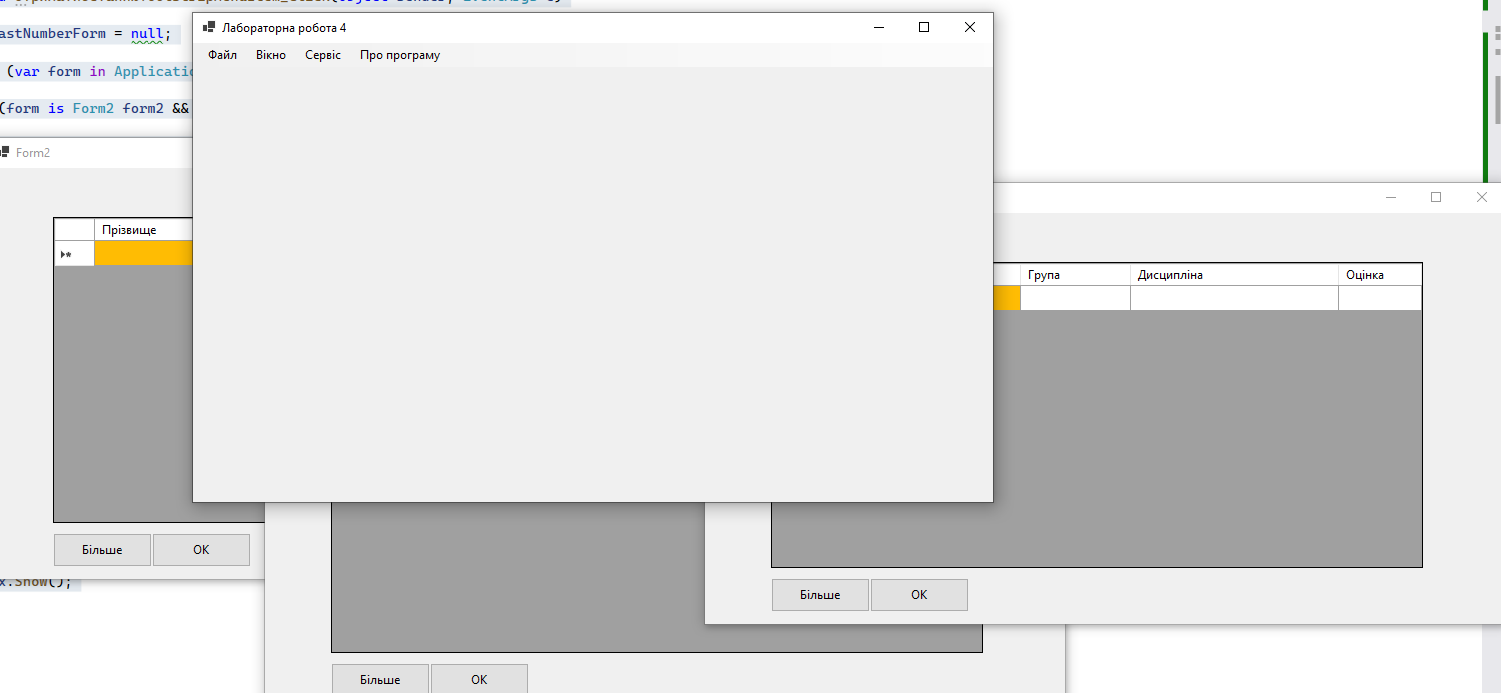
1. **Короткі теоретичні відомості про типи SDI та MDI інтерфейсів**

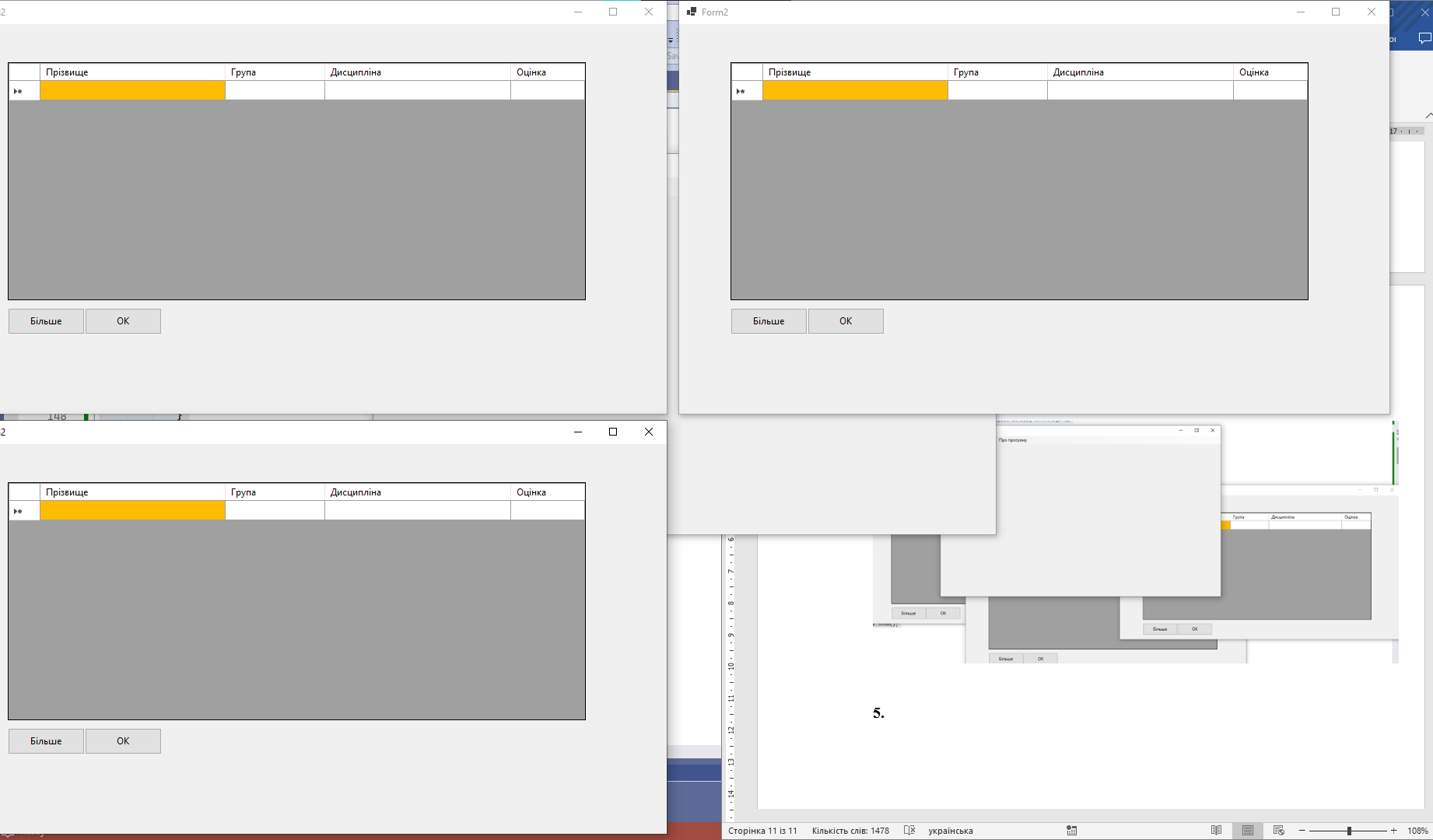
Інтерфейс створюваного додатка з незалежними вікнами називається SDI (Single Document Interface) – однодокументальний інтерфейс. Бувають додатки й з іншим типом інтерфейсу MDI (Multiple Document Interface) – багатодокументальний інтерфейс, у яких існує єдине головне ("батьківське") вікно, що містить дочірні вікна документів, як, наприклад, у текстовому процесорі Microsoft Word

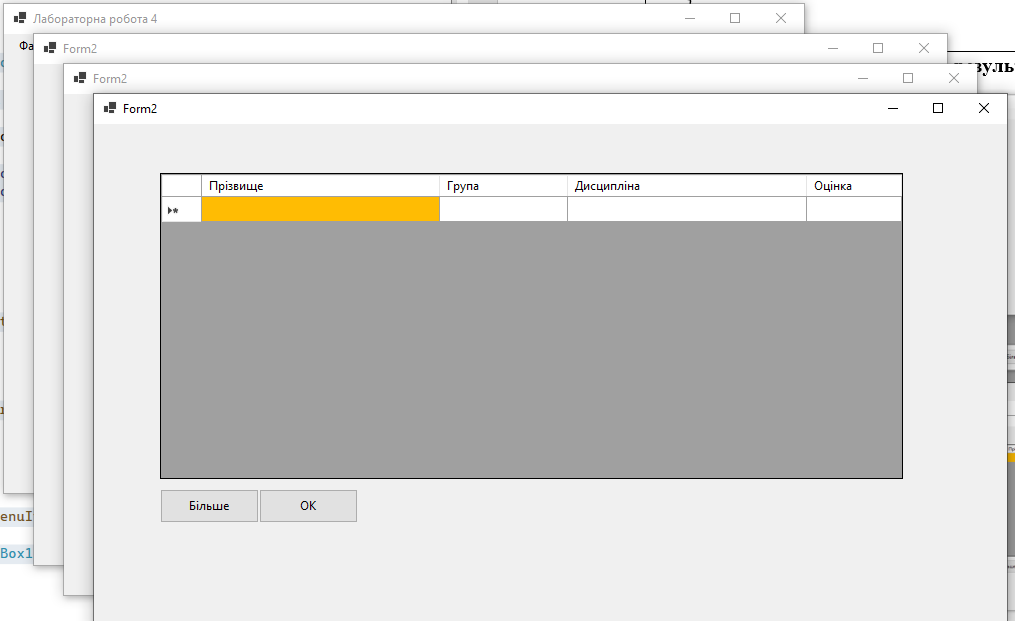
1. **Вихідний текст програми**

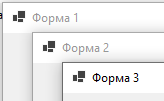
|  |
| --- |
| **Form1.cs**  using System.Windows.Forms;  using System.Linq;  using System.Windows.Forms.Layout;  namespace Lab4  {  public partial class Form1 : Form  {  private int formCounter = 1;  private int lastFormNumber = 0;  public Form1()  {  InitializeComponent();  }  private void створитиToolStripMenuItem\_Click(object sender, EventArgs e)  {  Form2 form2 = new Form2();  form2.Show();  }  private void мозаїкаToolStripMenuItem\_Click(object sender, EventArgs e)  {  var form2Windows = Application.OpenForms.OfType<Form2>();  TileWindows(form2Windows.Select(f => f.Handle).ToArray(), form2Windows.Count());  }  private const int SWP\_ASYNCWINDOWPOS = 0x4000;  private const int SWP\_DEFERERASE = 0x2000;  private const int SWP\_DRAWFRAME = 0x0020;  private const int SWP\_FRAMECHANGED = 0x0020;  private const int SWP\_NOACTIVATE = 0x0010;  private const int SWP\_NOCOPYBITS = 0x0100;  private const int SWP\_NOMOVE = 0x0002;  private const int SWP\_NOOWNERZORDER = 0x0200;  private const int SWP\_NOREDRAW = 0x0008;  private const int SWP\_NOREPOSITION = 0x0200;  private const int SWP\_NOSENDCHANGING = 0x0400;  private const int SWP\_NOSIZE = 0x0001;  private const int SWP\_NOZORDER = 0x0004;  private const int SWP\_SHOWWINDOW = 0x0040;  [System.Runtime.InteropServices.DllImport("user32.dll", SetLastError = true)]  private static extern bool SetWindowPos(IntPtr hWnd, IntPtr hWndInsertAfter, int x, int y, int cx, int cy, int uFlags);  private static void TileWindows(IntPtr[] windows, int count)  {  const int cols = 2; // Кількість стовпців в мозаїчному розташуванні  const int rows = 2; // Кількість рядків в мозаїчному розташуванні  int width = Screen.PrimaryScreen.WorkingArea.Width / cols;  int height = Screen.PrimaryScreen.WorkingArea.Height / rows;  for (int i = 0; i < count; i++)  {  int x = (i % cols) \* width;  int y = (i / cols) \* height;  SetWindowPos(windows[i], IntPtr.Zero, x, y, width, height, SWP\_ASYNCWINDOWPOS | SWP\_SHOWWINDOW);  }  }  private void каскадомToolStripMenuItem\_Click(object sender, EventArgs e)  {  CascadeForm2Windows();  }  private void CascadeForm2Windows()  {  var form2Windows = Application.OpenForms.OfType<Form2>(); // Всі відкриті вікна типу Form2  int offset = 30; // Відступ між кожним вікном при каскадному розташуванні  int startX = this.Location.X + offset;  int startY = this.Location.Y + offset;  int cascadeX = startX; // Встановлення стартового значення координат  int cascadeY = startY;  foreach (var form in form2Windows)  {  form.StartPosition = FormStartPosition.Manual;  form.Location = new Point(cascadeX, cascadeY);  cascadeX += offset; // Збільшення значення на відступ offset, щоб  cascadeY += offset; // запезпечити зсув між кожним вікном  form.Focus();  }  }  private void закритиВсеToolStripMenuItem\_Click(object sender, EventArgs e)  {  // Отримання всіх відкритих форм типу Form2 і  // перетворення результату на ToList()  var form2Windows = Application.OpenForms.OfType<Form2>().ToList();  // Проходимось по кожній формі Form2 і викликаємо метод Close(),  // щоб закрити форму  foreach (var form in form2Windows)  {  form.Close();  }  }  private void пронумеруватиФормиToolStripMenuItem\_Click(object sender, EventArgs e)  {  formCounter = 1;  foreach (var form in Application.OpenForms)  {  if (form is Form2 form2)  {  form2.Text = $"Форма {formCounter++}";  }  }  }  private void отриматиПершуToolStripMenuItem\_Click(object sender, EventArgs e)  {  foreach (var form in Application.OpenForms)  {  if (form is Form2 form2 && form2.Text == "Форма 1")  {  form2.BringToFront();  break;  }  }  }  private void отриматиОстаннюToolStripMenuItem\_Click(object sender, EventArgs e)  {  Form2 lastNumberForm = null;  foreach (var form in Application.OpenForms)  {  if (form is Form2 form2 && int.TryParse(form2.Text.Replace("Форма ", ""), out int formNumber))  {  if (formNumber > lastFormNumber)  {  lastFormNumber = formNumber;  lastNumberForm = form2;  }  }  }  if (lastNumberForm != null)  {  lastNumberForm.BringToFront();  }  }  private void завершитиРоботуToolStripMenuItem\_Click(object sender, EventArgs e)  {  Close();  }  private void проПрограмуToolStripMenuItem\_Click(object sender, EventArgs e)  {  AboutBox1 aboutBox = new AboutBox1();  aboutBox.Show();  }  }  } |
| **Form2.cs**  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;  namespace Lab4  {  public partial class Form2 : Form  {  private bool isExpanded = false; // Змінна для відстеження стану розширення форми  private Point originalButtonExpandLocation; // Початкова позиція кнопки  private Point originalButtonOKLocation; // Початкова позиція кнопки  public Form2()  {  InitializeComponent();  this.Resize += Form1\_Resize; // Підписка на подію Form.Resize  int buttonTop = dataGridView1.Top + dataGridView1.Height + 10; // Визначення вертикального положення кнопки  buttonExpand.Location = new Point(buttonExpand.Left, buttonTop); // Встановлення нової позиції кнопки  buttonOK.Location = new Point(buttonExpand.Right, buttonTop); // Встановлення нової позиції кнопки  originalButtonExpandLocation = buttonExpand.Location; // Зберегти початкову позицію кнопки  originalButtonOKLocation = buttonOK.Location; // Зберегти початкову позицію кнопки  this.Height = 450; // Встановлення вихідної висоти форми  panel1.Visible = false; // Приховування панелі  panel2.Visible = false; // Приховування панелі  }  private void dataGridView1\_CellContentClick(object sender, DataGridViewCellEventArgs e)  {  }  private void Form2\_Load(object sender, EventArgs e)  {  int desiredWidth = (int)(this.Width \* 0.8); // Обчислення 80% від ширини форми  dataGridView1.Width = desiredWidth; // Встановлення ширини сітки DataGridView  }  private void Form1\_Resize(object sender, EventArgs e)  {  int desiredWidth = (int)(this.Width \* 0.8); // Обчислення 80% від ширини форми  dataGridView1.Width = desiredWidth; // Встановлення ширини сітки DataGridView  }  private void buttonExpand\_Click(object sender, EventArgs e)  {  if (isExpanded)  {  // Зменшення розміру форми до вихідних значень  this.Height = 450; // Встановлення вихідної висоти форми  panel1.Visible = false; // Приховування панелі  panel2.Visible = false; // Приховування панелі  buttonExpand.Text = "Більше"; // Зміна тексту кнопки на "Більше"  // Переміщення кнопок на початкову позицію  buttonExpand.Location = originalButtonExpandLocation;  buttonOK.Location = originalButtonOKLocation;  isExpanded = false; // Встановлення прапорця розширення в "вимкнено"  }  else  {  // Збільшення розміру форми та відкриття панелі  this.Height = 480; // Встановлення нової висоти форми  panel1.Visible = true; // Відображення панелі  panel2.Visible = true; // Відображення панелі  buttonExpand.Text = "Менше"; // Зміна тексту кнопки на "Менше"  // Зміщення кнопки вниз на висоту панелі  buttonExpand.Location = new Point(originalButtonExpandLocation.X, originalButtonExpandLocation.Y + panel1.Height);  buttonOK.Location = new Point(originalButtonOKLocation.X, originalButtonOKLocation.Y + panel1.Height);  isExpanded = true; // Встановлення прапорця розширення в "увімкнено"  }  }  private void buttonOK\_Click(object sender, EventArgs e)  {  if (radioSave.Checked)  {  SaveSelectedRowsToFile("H:\\University\\2 курс\\2 семестр\\ООП\\Код\\Lab4\\selected\_rows.txt");  }  else if (radioView.Checked)  {  ConfigureGridViewAppearance();  }  }  private void SaveSelectedRowsToFile(string filePath)  {  // Відкриття файлу для запису  using (StreamWriter writer = new StreamWriter(filePath))  {  // Проходження по кожному рядку таблиці  foreach (DataGridViewRow row in dataGridView1.Rows)  {  // Перевірка, чи рядок є виділеним  if (row.Selected)  {  // Збереження значення рядка у файл  for (int j = 0; j < dataGridView1.Columns.Count; j++)  {  string value = row.Cells[j].Value?.ToString() ?? string.Empty;  writer.Write(value);  if (j < dataGridView1.Columns.Count - 1)  {  writer.Write(" ");  }  }  writer.WriteLine();  }  }  }  // Повідомлення про успішне збереження  MessageBox.Show("Рядки були успішно збережені у файл.");  }  private void ConfigureGridViewAppearance()  {  // Задати вікну білий колір  this.BackColor = Color.White;  // Задати межам сітки в dataGridView1 чорний колір  dataGridView1.GridColor = Color.Black;  // Зменшити вертикальний розмір dataGridView1 до розмірів сітки  dataGridView1.Height = dataGridView1.Rows.GetRowsHeight(DataGridViewElementStates.Visible) + dataGridView1.ColumnHeadersHeight;  }  private void checkBox1\_CheckedChanged(object sender, EventArgs e)  {  SelectDataGridViewRow(0, checkBox1.Checked);  }  private void checkBox2\_CheckedChanged(object sender, EventArgs e)  {  SelectDataGridViewRow(1, checkBox2.Checked);  }  private void checkBox3\_CheckedChanged(object sender, EventArgs e)  {  SelectDataGridViewRow(2, checkBox3.Checked);  }  private void checkBox4\_CheckedChanged(object sender, EventArgs e)  {  SelectDataGridViewRow(3, checkBox4.Checked);  }  private void checkBox5\_CheckedChanged(object sender, EventArgs e)  {  SelectDataGridViewRow(4, checkBox5.Checked);  }  private void SelectDataGridViewRow(int rowIndex, bool isSelected)  {  if (rowIndex >= 0 && rowIndex < dataGridView1.Rows.Count)  {  dataGridView1.Rows[rowIndex].Selected = isSelected;  }  }  }  } |
| **AboutBox1.cs**  using System;  using System.Collections.Generic;  using System.ComponentModel;  using System.Drawing;  using System.Linq;  using System.Reflection;  using System.Threading.Tasks;  using System.Windows.Forms;  namespace Lab4  {  partial class AboutBox1 : Form  {  public AboutBox1()  {  InitializeComponent();  this.Text = "Про програму";  this.labelProductName.Text = "Лабораторна робота 4";  this.labelVersion.Text = String.Format("Version {0}", AssemblyVersion);  this.labelCopyright.Text = AssemblyCopyright;  this.labelCompanyName.Text = "Rostik Hubariev";  }  #region Assembly Attribute Accessors  public string AssemblyTitle  {  get  {  object[] attributes = Assembly.GetExecutingAssembly().GetCustomAttributes(typeof(AssemblyTitleAttribute), false);  if (attributes.Length > 0)  {  AssemblyTitleAttribute titleAttribute = (AssemblyTitleAttribute)attributes[0];  if (titleAttribute.Title != "")  {  return titleAttribute.Title;  }  }  return System.IO.Path.GetFileNameWithoutExtension(Assembly.GetExecutingAssembly().CodeBase);  }  }  public string AssemblyVersion  {  get  {  return Assembly.GetExecutingAssembly().GetName().Version.ToString();  }  }  public string AssemblyDescription  {  get  {  object[] attributes = Assembly.GetExecutingAssembly().GetCustomAttributes(typeof(AssemblyDescriptionAttribute), false);  if (attributes.Length == 0)  {  return "";  }  return ((AssemblyDescriptionAttribute)attributes[0]).Description;  }  }  public string AssemblyProduct  {  get  {  object[] attributes = Assembly.GetExecutingAssembly().GetCustomAttributes(typeof(AssemblyProductAttribute), false);  if (attributes.Length == 0)  {  return "";  }  return ((AssemblyProductAttribute)attributes[0]).Product;  }  }  public string AssemblyCopyright  {  get  {  object[] attributes = Assembly.GetExecutingAssembly().GetCustomAttributes(typeof(AssemblyCopyrightAttribute), false);  if (attributes.Length == 0)  {  return "";  }  return ((AssemblyCopyrightAttribute)attributes[0]).Copyright;  }  }  public string AssemblyCompany  {  get  {  object[] attributes = Assembly.GetExecutingAssembly().GetCustomAttributes(typeof(AssemblyCompanyAttribute), false);  if (attributes.Length == 0)  {  return "";  }  return ((AssemblyCompanyAttribute)attributes[0]).Company;  }  }  #endregion  private void AboutBox1\_Load(object sender, EventArgs e)  {  }  private void okButton\_Click\_1(object sender, EventArgs e)  {  Close();  }  }  } |

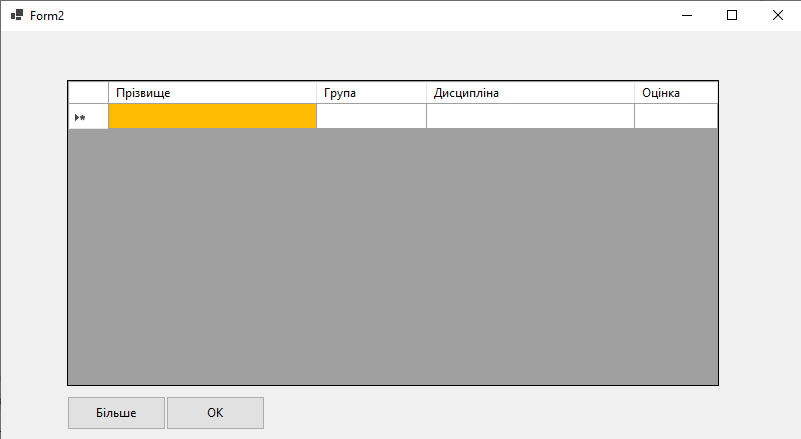
1. **Знімок інтерфейсу програми з результатами роботи**

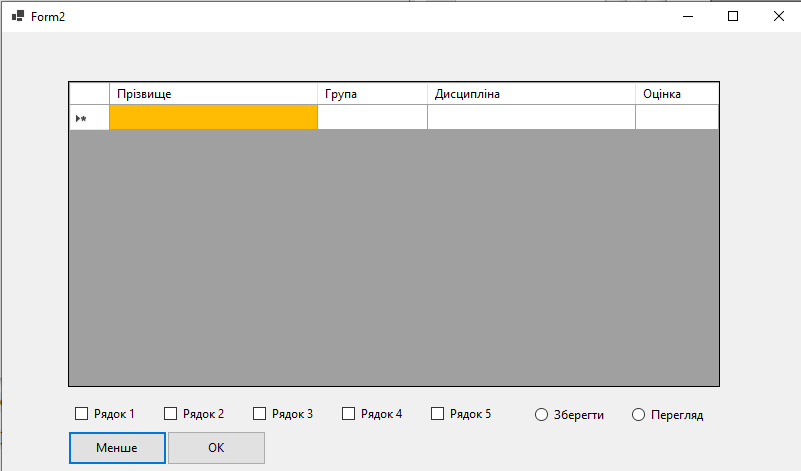
****

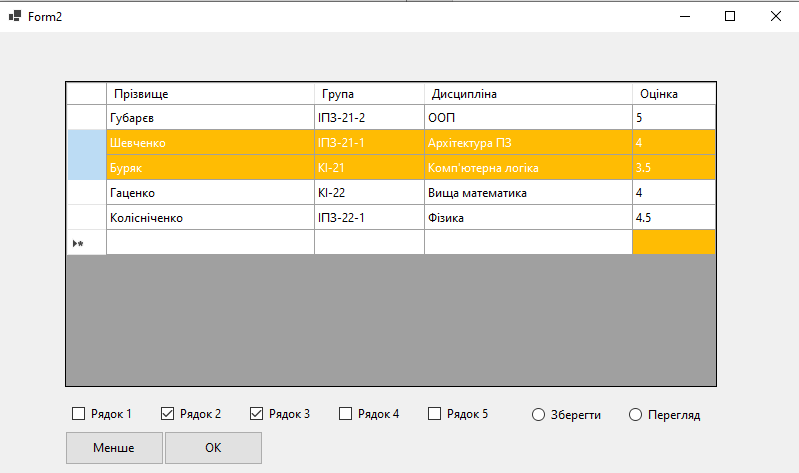
****

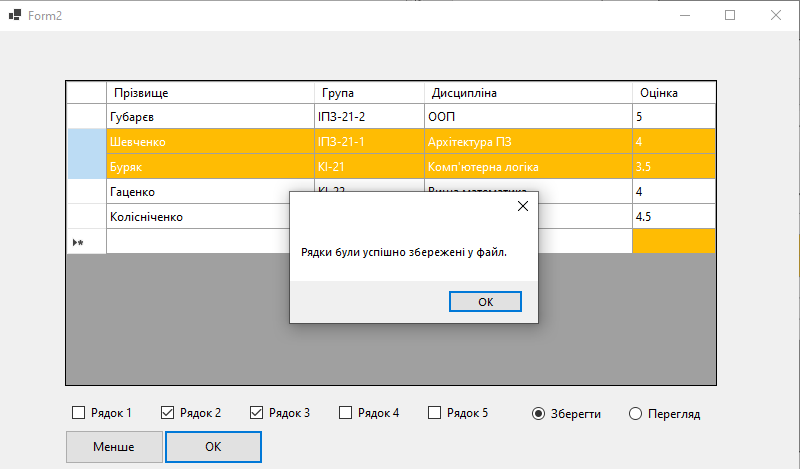
****

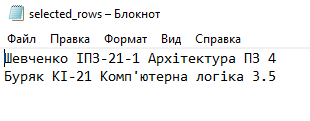
****

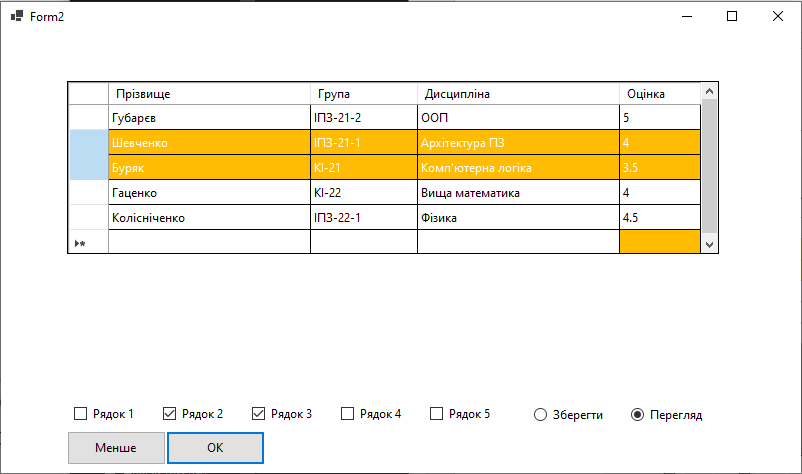
****

****

****

****

****

****

1. **Короткі висновки**

В цій лабораторній роботі я навчився працювати з декількома формами, дізнався про типи SDI та MDI інтерфейсів, а також навчився змінювати розмір вікна та змінювати положення об’єктів.

1. **Список літератури**

* [**https://learn.microsoft.com/ru-ru/dotnet/desktop/winforms/controls/windows-forms-controls-by-function?view=netframeworkdesktop-4.8**](https://learn.microsoft.com/ru-ru/dotnet/desktop/winforms/controls/windows-forms-controls-by-function?view=netframeworkdesktop-4.8)
* [**http://lib.kart.edu.ua/bitstream/123456789/6106/1/%D0%9A%D0%BE%D0%BD%D1%81%D0%BF%D0%B5%D0%BA%D1%82%20%D0%BB%D0%B5%D0%BA%D1%86%D1%96%D0%B9.pdf**](http://lib.kart.edu.ua/bitstream/123456789/6106/1/%D0%9A%D0%BE%D0%BD%D1%81%D0%BF%D0%B5%D0%BA%D1%82%20%D0%BB%D0%B5%D0%BA%D1%86%D1%96%D0%B9.pdf)