**ЛАБОРАТОРНА РОБОТА № 13**

**Мета:** вивчити принципи реалізації перевантажених операцій у мові

C#; навчитися реалізовувати власні класи, які дозволяють виконувати операції

над об’єктами класу.

**Хід роботи:**

**Завдання 1.** У віконному додатку реалізувати програму, яка дозволить:

- вибирати користувачу довільні файли для подальшої їх обробки;

- шифруватиме вибрані файли;

- архівуватиме вибрані файли;

- розархівування вибрані файли;

- розпаковуватиме вибрані файли.

**Лістеннінг класів:**

public class GzarArchiver

{

private const string TempFileName = "fileinfo.txt";

private static string tempFilePath = Path.Combine(Path.GetTempPath(), TempFileName);

public static void CreateArchive(ListBox listBox)

{

List<string> files = new List<string>();

foreach (string file in listBox.Items)

{

if (file.EndsWith(".gzar"))

{

continue;

}

if (file.Contains(".crypt"))

{

files.Add(file);

files.Add(file.Replace(".crypt", ".temp"));

continue;

}

files.Add(file);

}

if(files.Count == 0)

{

return;

}

using (StreamWriter writer = new StreamWriter(tempFilePath))

{

foreach (string file in files)

{

writer.WriteLine(file);

}

}

string zipFilePath = Path.Combine("E:\\Унік\\1 Курс\\Лаби ООП\\lab\_13\\archives\\", Path.GetFileNameWithoutExtension(files[0]) + ".zip");

using (FileStream archiveStream = new FileStream(zipFilePath, FileMode.Create))

{

using (ZipArchive archive = new ZipArchive(archiveStream, ZipArchiveMode.Create))

{

archive.CreateEntryFromFile(tempFilePath, TempFileName);

foreach (string file in files)

{

string entryName = Path.GetFileName(file);

archive.CreateEntryFromFile(file, entryName);

}

}

}

File.Move(zipFilePath, Path.ChangeExtension(zipFilePath, ".gzar"));

File.Delete(tempFilePath);

foreach (string file in files)

{

File.Delete(file);

}

listBox.Items.Clear();

listBox.Items.Add(Path.ChangeExtension(zipFilePath, ".gzar"));

}

public static void ExtractArchive(ListBox listBox)

{

List<string> oldFiles = new List<string>();

List<string> newFiles = new List<string>();

foreach (string archivePath in listBox.Items)

{

if (archivePath.EndsWith(".gzar"))

{

using (FileStream archiveStream = new FileStream(archivePath, FileMode.Open))

using (ZipArchive archive = new ZipArchive(archiveStream, ZipArchiveMode.Read))

{

var tempEntry = archive.GetEntry(TempFileName);

tempEntry.ExtractToFile(tempFilePath, true);

List<string> fileMappings = new List<string>();

using (StreamReader reader = new StreamReader(tempFilePath))

{

string line;

while ((line = reader.ReadLine()) != null)

{

fileMappings.Add(line);

}

}

foreach (string filePath in fileMappings)

{

string fileName = Path.GetFileName(filePath);

var fileEntry = archive.GetEntry(fileName);

string destinationPath = filePath;

fileEntry.ExtractToFile(destinationPath, true);

if(filePath.EndsWith(".crypt"))

newFiles.Add(filePath);

}

}

File.Delete(tempFilePath);

File.Delete(archivePath);

}

else

oldFiles.Add(archivePath);

}

listBox.Items.Clear();

if (oldFiles.Count != 0)

foreach (string file in oldFiles)

listBox.Items.Add(file);

foreach (string file in newFiles)

listBox.Items.Add(file);

}

}

public class XOREncryption

{

public static void ProcessFiles(ListBox listBox)

{

foreach (string file in listBox.Items)

{

if (!file.Contains(".crypt"))

ProcessFile(file);

}

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

{

if (!listBox.Items[i].ToString().Contains(".crypt"))

listBox.Items[i] += ".crypt";

}

}

public static void DecodeFiles(ListBox listBox)

{

foreach (string file in listBox.Items)

{

if (file.Contains(".crypt"))

DecodeFile(file);

}

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

{

if (listBox.Items[i].ToString().Contains(".crypt"))

listBox.Items[i] = listBox.Items[i].ToString().Replace(".crypt", "");

}

}

public static byte[] ReadFile(string path)

{

return File.ReadAllBytes(path);

}

public static byte[] XORencode(byte[] data, byte[] key)

{

byte[] result = new byte[data.Length];

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

{

result[i] = (byte)(data[i] ^ key[i % key.Length]);

}

return result;

}

public static void SaveFile(string path, byte[] data, byte[] key)

{

string encryptedFilePath = path + ".crypt";

string keyFilePath = path + ".temp";

File.WriteAllBytes(encryptedFilePath, data);

File.WriteAllBytes(keyFilePath, key);

File.Delete(path);

}

public static byte[] GenerateKey(int length)

{

byte[] key = new byte[256];

Random random = new Random();

random.NextBytes(key);

return key;

}

public static void ProcessFile(string path)

{

byte[] data = ReadFile(path);

byte[] key = GenerateKey(data.Length);

byte[] encodedData = XORencode(data, key);

SaveFile(path, encodedData, key);

}

public static void DecodeFile(string encryptedFilePath)

{

string keyFilePath = encryptedFilePath.Replace(".crypt", ".temp");

string originalFilePath = encryptedFilePath.Replace(".crypt", "");

byte[] encryptedData = ReadFile(encryptedFilePath);

byte[] key = ReadFile(keyFilePath);

byte[] decodedData = XORencode(encryptedData, key);

File.WriteAllBytes(originalFilePath, decodedData);

File.Delete(encryptedFilePath);

File.Delete(keyFilePath);

}

}

**Форма:**

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

listBox1.AllowDrop = true;

listBox1.DragDrop += new DragEventHandler(listBox1\_DragDrop);

listBox1.DragEnter += new DragEventHandler(listBox1\_DragEnter);

}

private void listBox1\_DragEnter(object sender, DragEventArgs e)

{

if (e.Data.GetDataPresent(DataFormats.FileDrop))

{

e.Effect = DragDropEffects.Copy;

}

else

{

e.Effect = DragDropEffects.None;

}

}

private void listBox1\_DragDrop(object sender, DragEventArgs e)

{

string[] files = (string[])e.Data.GetData(DataFormats.FileDrop);

foreach (string file in files)

{

if (!listBox1.Items.Contains(file))

{

listBox1.Items.Add(file);

}

}

}

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

{

GzarArchiver.CreateArchive(listBox1);

}

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

{

GzarArchiver.ExtractArchive(listBox1);

}

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

{

listBox1.Items.Clear();

}

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

{

OpenFileDialog openFileDialog1 = new OpenFileDialog();

openFileDialog1.InitialDirectory = "c:\\";

openFileDialog1.Filter = "Text files (\*.txt)|\*.txt|All files (\*.\*)|\*.\*";

openFileDialog1.FilterIndex = 2;

openFileDialog1.Multiselect = true;

if (openFileDialog1.ShowDialog() == DialogResult.OK)

{

foreach (string file in openFileDialog1.FileNames)

{

if (!listBox1.Items.Contains(file))

{

listBox1.Items.Add(file);

}

}

}

}

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

{

if (listBox1.SelectedItem != null)

{

int selectedIndex = listBox1.SelectedIndex;

listBox1.Items.RemoveAt(selectedIndex);

}

}

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

{

XOREncryption.ProcessFiles(listBox1);

}

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

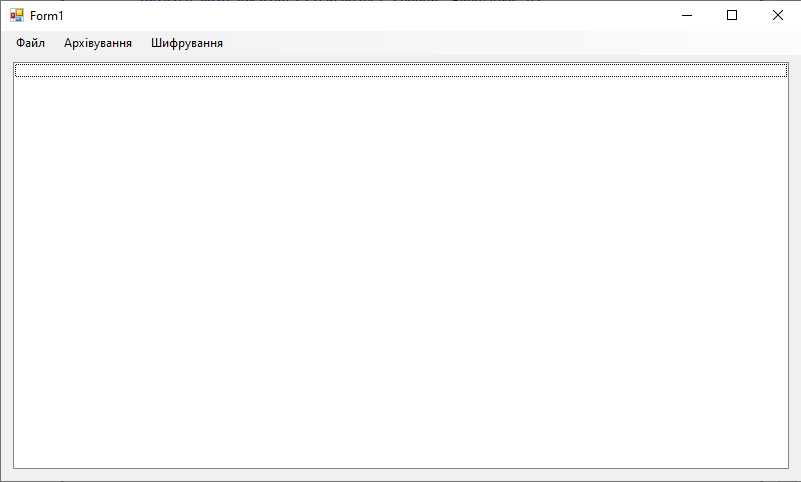
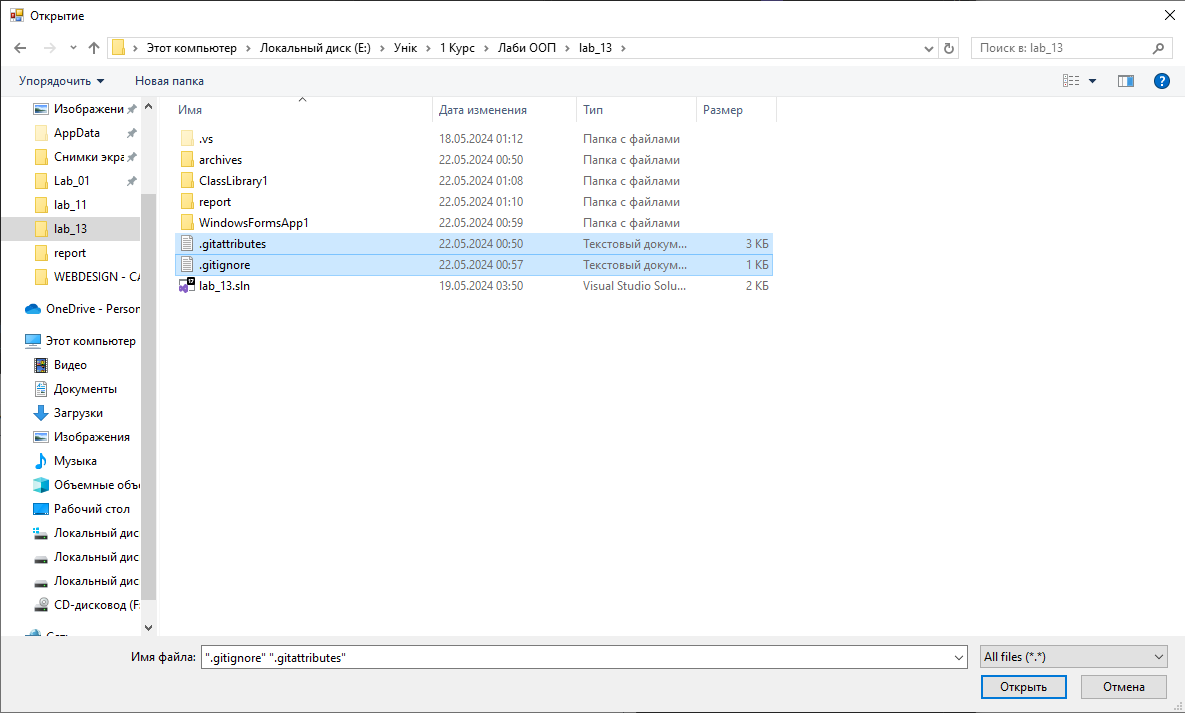
{

XOREncryption.DecodeFiles(listBox1);

}

}

**Приклад виконання:**

****

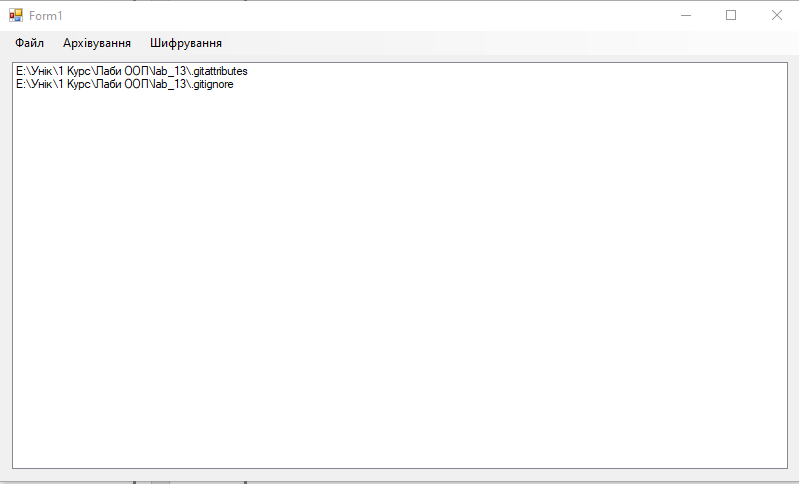
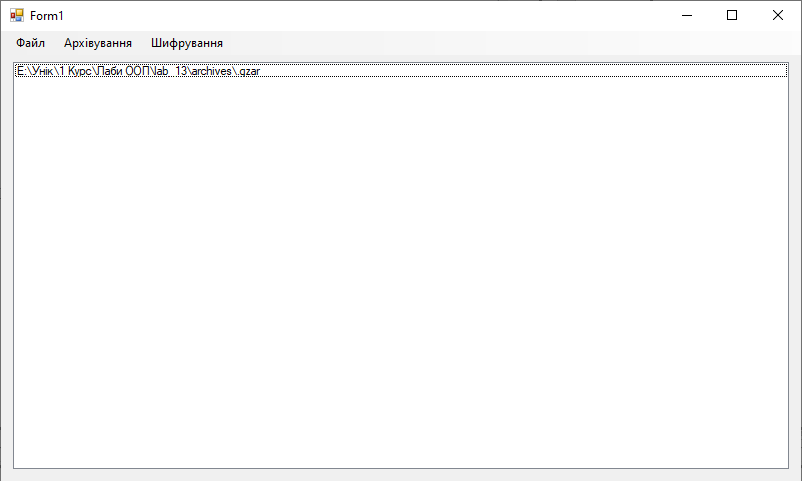
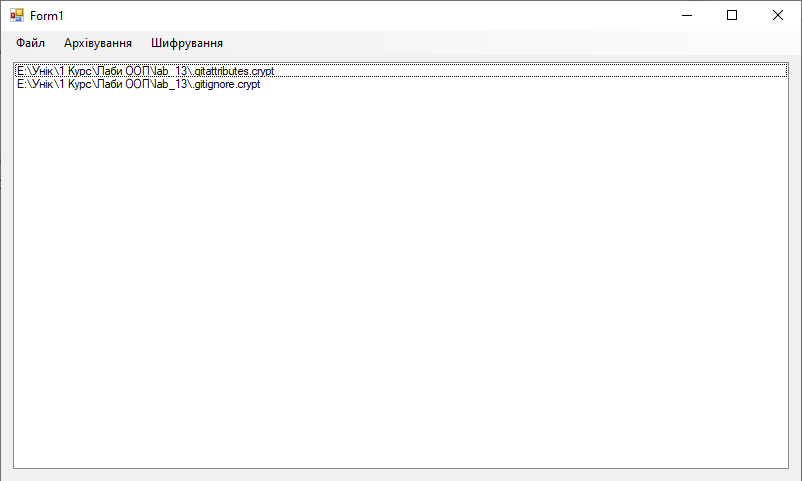
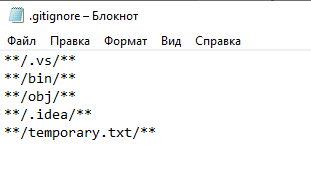
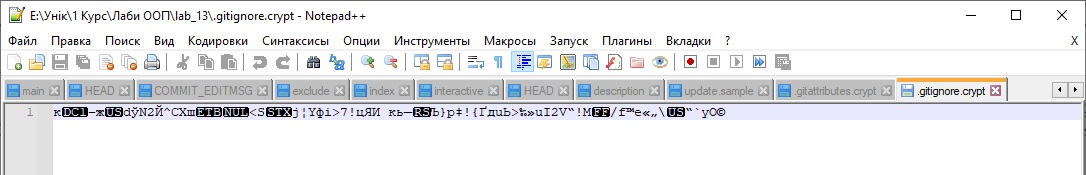
**  **  ** **

Рис. 1, 2, 3, 4, 5, 6, 7, 8. Приклад роботи додатку.

**Висновок:** На даній лабораторній роботі я вивчив принципи реалізації перевантажених операцій у мові C# навчився реалізовувати власні класи, які дозволяють виконувати операції над об’єктами класу.