**Практична робота №6**

**ВИКОРИСТАННЯ СОМ-ТЕХНОЛОГІЇ.**

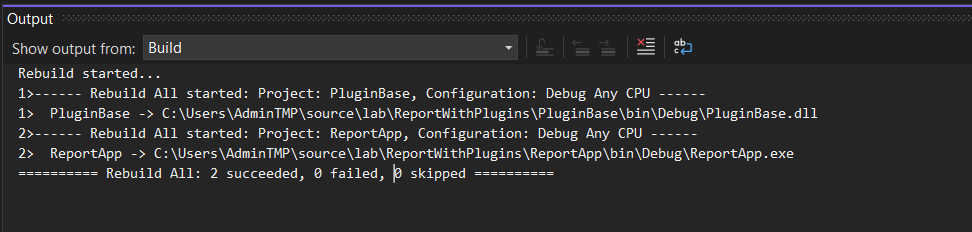
**ЕКСПОРТ ДАНИХ У ДОКУМЕНТИ WORD ТА EXCEL**

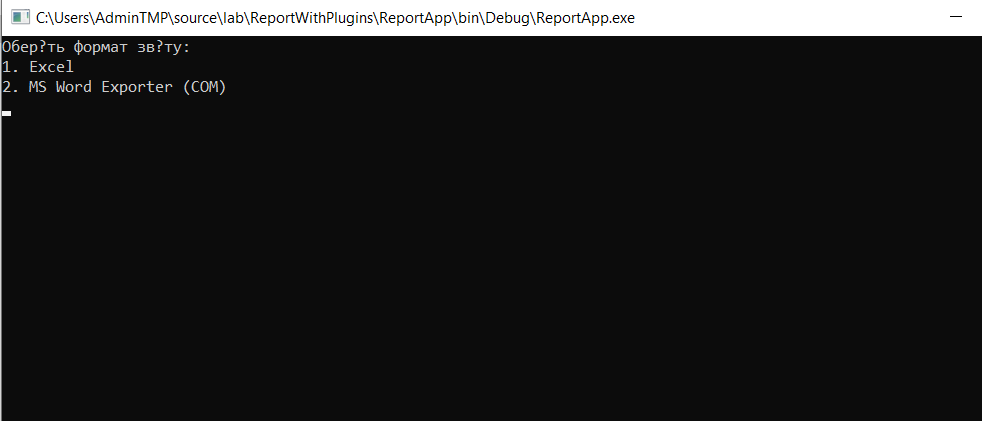
Мета роботи : Вивчення та практичне відпрацювання механізмів використання компонентів на основі СОМ -технології та її застосування в .NET. Вивчення особливостей використання додатків та програмного формування документів MS Office.

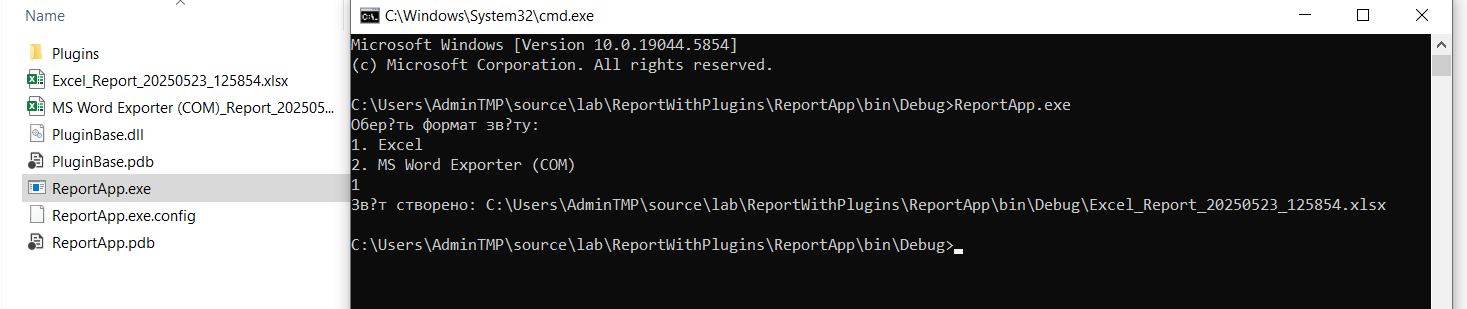
Завдання

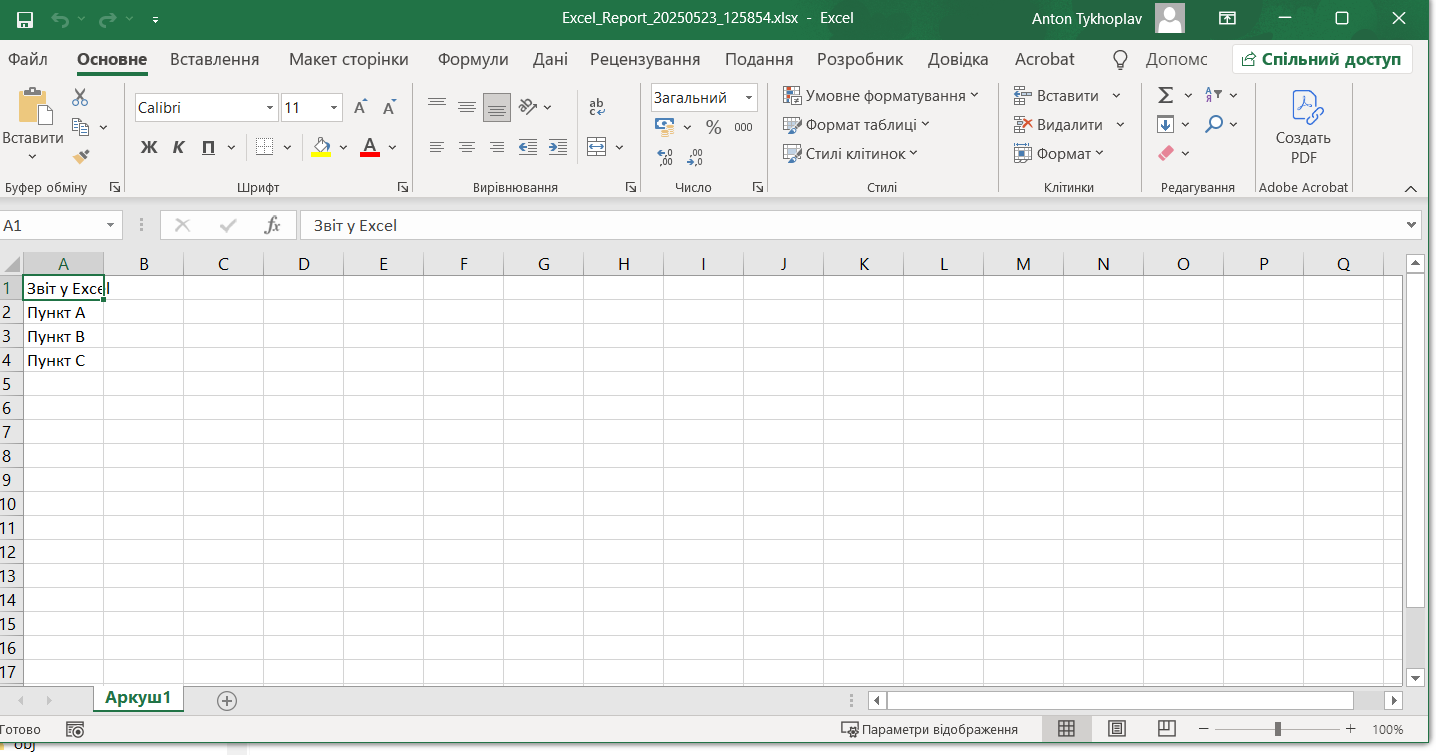
Реалізувати у практичній роботі генерацію звітів у форматах MS Word та MS Excel за допомогою COM-об’єктів Microsoft Office [1, 2]. Структуру звіту (таблиця, заголовок, підпис тощо) слід формувати динамічно відповідно до наповнення предметної області (див. Варіанти завдання). У результаті користувач повинен мати можливість отримати (або завантажити у випадку WebUI або Tg Bot) сформовані документи у форматах .docx та .xlsx, які створюються програмним застосунком.

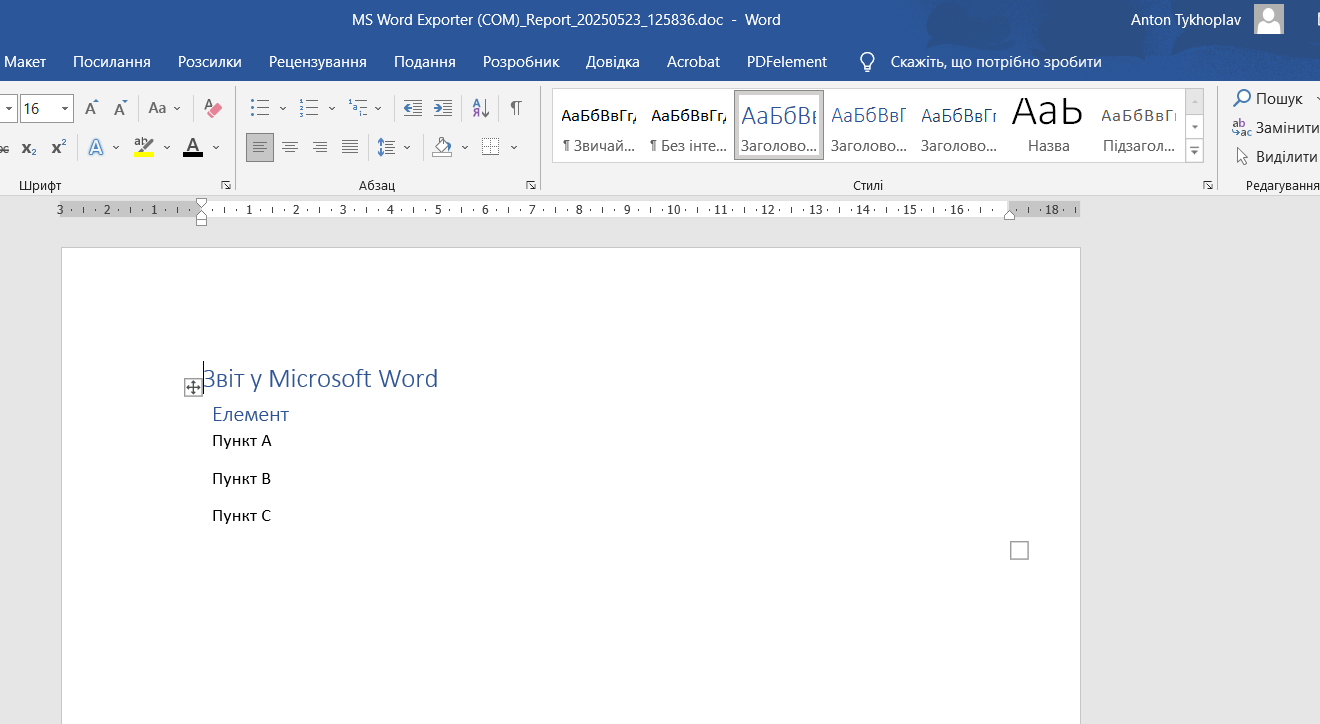
У межах додаткового завдання необхідно реалізувати два незалежні plug-in модулі (dll-assembly) для формування звітів у форматах MS Word і MS Excel. Під plug-in мається на увазі динамічне завантаження модулів під час виконання основної програми, а не під час компіляції. Для реалізації додаткової частини завдання (п. 5, таблиця 6.1) слід застосувати методи пізнього зв'язування та завантажувати модулі формування звітів за бажанням користувача. Основи динамічного зв'язування розглянуто у [3–5].











using System;

using System.IO;

using System.Linq;

using System.Reflection;

using PluginBase;

namespace ReportApp

{

class Program

{

static void Main()

{

var pluginDir = Path.Combine(AppContext.BaseDirectory, "Plugins");

var dlls = Directory.GetFiles(pluginDir, "\*.dll");

var generators = dlls

.SelectMany(file =>

{

var asm = Assembly.LoadFrom(file);

return asm.GetTypes()

.Where(t => typeof(IReportGenerator).IsAssignableFrom(t) && !t.IsInterface)

.Select(t => (IReportGenerator)Activator.CreateInstance(t)!);

})

.ToList();

Console.WriteLine("Оберіть формат звіту:");

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

Console.WriteLine($"{i + 1}. {generators[i].Name}");

var idx = int.Parse(Console.ReadLine()!) - 1;

var gen = generators[idx];

var data = new[] { "Пункт A", "Пункт B", "Пункт C" };

var fileName = $"{gen.Name}\_Report\_{DateTime.Now:yyyyMMdd\_HHmmss}" +

(gen.Name == "Word" ? ".docx" : ".xlsx");

var output = Path.Combine(AppContext.BaseDirectory, fileName);

gen.GenerateReport(data, output);

Console.WriteLine($"Звіт створено: {output}");

}

}

}

using System;

using System.Runtime.InteropServices;

using PluginBase;

using Excel = Microsoft.Office.Interop.Excel;

namespace ExcelPlugin

{

public class ExcelReportGenerator : IReportGenerator

{

public string Name => "Excel";

public void GenerateReport(string[] items, string outputPath)

{

var app = new Excel.Application();

var wb = app.Workbooks.Add();

var ws = (Excel.Worksheet)wb.Worksheets[1];

ws.Cells[1, 1] = "Звіт у Excel";

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

ws.Cells[i + 2, 1] = items[i];

wb.SaveAs(outputPath);

wb.Close();

app.Quit();

// COM cleanup

Marshal.ReleaseComObject(ws);

Marshal.ReleaseComObject(wb);

Marshal.ReleaseComObject(app);

GC.Collect();

GC.WaitForPendingFinalizers();

}

}

}

// Файл: WordPlugin/WordReportGenerator.cs

using System;

using System.Runtime.InteropServices;

using PluginBase;

using Word = Microsoft.Office.Interop.Word;

namespace WordPlugin

{

public class WordReportGenerator : IReportGenerator

{

public string Name => "MS Word Exporter (COM)";

public void GenerateReport(string[] items, string outputPath)

{

var wordApp = new Word.Application();

Word.Document doc = null;

try

{

// Створюємо новий документ

doc = wordApp.Documents.Add();

// Додаємо заголовок

Word.Paragraph heading = doc.Paragraphs.Add();

heading.Range.Text = "Звіт у Microsoft Word";

heading.Range.set\_Style(Word.WdBuiltinStyle.wdStyleHeading1);

heading.Range.InsertParagraphAfter();

// Готуємо діапазон для таблиці — останній параграф документа

Word.Range tableRange = doc.Paragraphs[doc.Paragraphs.Count].Range;

// Додаємо таблицю: (кількість рядків = items.Length + 1, 1 стовпець)

Word.Table table = doc.Tables.Add(tableRange, items.Length + 1, 1);

table.Cell(1, 1).Range.Text = "Елемент";

table.Rows[1].Range.set\_Style(Word.WdBuiltinStyle.wdStyleHeading2);

// Заповнюємо таблицю даними

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

{

table.Cell(i + 2, 1).Range.Text = items[i];

}

// Зберігаємо документ

doc.SaveAs2(outputPath);

}

finally

{

// Правильне закриття та звільнення COM-об'єктів

if (doc != null)

{

doc.Close(Word.WdSaveOptions.wdDoNotSaveChanges);

Marshal.ReleaseComObject(doc);

}

wordApp.Quit();

Marshal.ReleaseComObject(wordApp);

GC.Collect();

GC.WaitForPendingFinalizers();

}

}

}

}