

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

**Український державний університет науки і технологій**

Кафедра «Комп’ютерні інформаційні технології»

**Лабораторна робота №2**

**з дисципліни «Професійна практика програмної інженерії»**

## на тему:

**«Реалізація проекту у відповідності до розроблених специфікацій.»**

Виконав:

студент гр. ПЗ2011

Кулик Сергій Вадимович

Прийняла:

Шаповал І.В.

Дніпро, 2022

**Тема.** Реалізація проекту у відповідності до розроблених специфікацій.

**Мета.** Отримання навичок щодо вибору мови програмування, командної розробки та оцінки прийнятих рішень на початкових етапах.

**Постановка задачі**

Дана лабараторна робота складається з файлу бібліотеки .dll

**Завдання**

Бібліотека для роботи з рядками.

**Обґрунтування вибору мови програмування**

Для розгляду було обрано наступні мови програмування: Java, C++, C#

**Java**

1. ***Платформа розробки (область застосування)***

Бібліотека роботи з матрицями може застосовуватися у проектах будь-якої складності, починаючи від програм початківців, до складних систем.

1. ***Гнучкість***

У редагуванні написаного коду бібліотеки немає потреби.

1. ***Тривалість розробки***

Для розробки бібліотеки за допомогою мови Java необхідно небагато часу. Для написання реалізації бібліотеки значної кількості коду не потрібно. Основна відмінність з іншими мовами програмування – робота здійснюється у класах, що є особливість мови Java.

1. ***Продуктивність***

Поставлена задача досить проста, а методи, які в ній реалізовані, виконують поставлені перед ними задачі. При рішенні завдання методами зазначеної бібліотеки, швидкість виконання операції залежить від обраної функції, та розміру масиву. Продуктивність функцій самої бібліотеки майже не залежить від обраної (з виділених трьох) мови програмування, адже реалізація у них буде майже однакова.

1. ***Співтовариство***

Мова знаходиться у топ-3 по популярності, велика кількість створених бібліотек, більшість необхідних задач які необхідно виконати уже є реалізованими іншими розробниками.

**C++**

1. ***Платформа розробки (область застосування)***

Бібліотека роботи з матрицями може застосовуватися у проектах будь-якої складності, починаючи від програм початківців, до складних систем.

1. ***Гнучкість***

Написання бібліотеки мовою С++ дає можливість контролювати пам’ять при роботі з масивами. Проте особливих маніпуляцій з пам’яттю немає необхідності робити.

1. ***Тривалість розробки***

Для розробки бібліотеки за допомогою мови С++ необхідно витратити деякий час для налаштування самого проекту бібліотеки. При створенні проекту автоматично створюються усі необхідні файли для роботи бібліотеки, а для написання реалізації бібліотеки значної кількості коду не потрібно.

1. ***Продуктивність***

Поставлена задача досить проста, а методи, які в ній реалізовані, виконують поставлені перед ними задачі. При рішенні завдання методами зазначеної бібліотеки, швидкість виконання операції залежить від обраної функції, та розміру масиву. Продуктивність функцій самої бібліотеки майже не залежить від обраної (з виділених трьох) мови програмування, адже реалізація у них буде майже однакова.

1. ***Співтовариство***

На даній мові ведуть розробку велика кількість людей, існує велика кількість різних проектів та рішень різноманітних завдань. Також наявна велика кількість інформації, про те, як можна написати бібліотеку, велика кількість алгоритмів(наприклад множення матриць або пошуку детермінанта).

**C#**

1. ***Платформа розробки (область застосування)***

Бібліотека роботи з матрицями може застосовуватися у проектах будь-якої складності, починаючи від програм початківців, до складних систем.

1. ***Гнучкість***

У редагуванні написаного коду бібліотеки немає потреби.

1. ***Тривалість розробки***

При створенні проекту на C# автоматично створюються усі необхідні файли для роботи бібліотеки, а для написання реалізації бібліотеки значної кількості коду не потрібно. Так як і мова Java, C# має об’єкто-орієнтовану структуру.

1. ***Продуктивність***

Поставлена задача досить проста, а методи, які в ній реалізовані, виконують поставлені перед ними задачі. При рішенні завдання методами зазначеної бібліотеки, швидкість виконання операції залежить від обраної функції, та розміру масиву. Продуктивність функцій самої бібліотеки майже не залежить від обраної (з виділених трьох) мови програмування, адже реалізація у них буде майже однакова.

1. ***Співтовариство***

На даній мові ведуть розробку велика кількість людей, існує велика кількість різних проектів та рішень різноманітних завдань. Також наявна велика кількість інформації, про те, як можна написати бібліотеку, велика кількість алгоритмів(наприклад множення матриць або пошуку детермінанта).

Кожна з мов є гарним варіантом для написання бібліотеки. С++ - є гарним варіантом для написання бібліотеки, проте я віддаю перевагу мові програмуванню C#, оскільки маю більше досвіду у роботі з ним.

Порівнявши всі плюси та мінуси створення бібліотеки на тій чи іншій мові програмування було складено наступний список за пріорітетністю:

1) С#

2) Java

3) C++

**Текст програми**

namespace StringLibrary

{

public class String

{

/// <summary>

/// Приєднує string1 в кінець string2

/// </summary>

/// <param name="string1">Рядок до якого відбудеться приєднання</param>

/// <param name="string2">Рядок, який приєднається</param>

/// <returns>Рядок, який об'єднує в собі перший і другий рядок</returns>

public static string Concat(string string1, string string2)

{

char[] mass = new char[string1.Length + string2.Length];

for (int i = 0; i < string1.Length; i++)

{

mass[i] = string1[i];

}

for (int i = 0; i < string2.Length; i++)

{

mass[string1.Length + i] = string2[i];

}

return new string(mass);

}

/// <summary>

/// Шукає перше входження підрядка в рядок

/// </summary>

/// <param name="string1">Рядок в якому відбувається пошук</param>

/// <param name="string2">Рядок, який необхідно знайти</param>

/// <returns>

/// Індекс першого символа знайденого підрядка в рядку

/// -1, якщо підрядка не знайдено

/// </returns>

public static int IndexOf(string string1, string string2)

{

int index = -1;

for (int i = 0; i < string1.Length; i++)

{

if (string1[i] == string2[0])

{

bool flag = true;

for (int j = 0; j < string2.Length; j++)

{

if (string1[i + j] != string2[j])

{

flag = false;

break;

}

}

if (flag)

{

index = i;

break;

}

}

}

return index;

}

/// <summary>

/// Вставляє підрядок в рядок в указану індексом позицію

/// </summary>

/// <param name="string1">Рядок для вставки</param>

/// <param name="string2">Рядок, який буде вставлено</param>

/// <param name="index">Позиція в яку потрібно вставити рядок</param>

/// <returns>Рядок зі вставленим підрядком в позицію за індексом</returns>

/// <exception cref="IndexOutOfRangeException"></exception>

public static string Insert(string string1, string string2, int index)

{

if (index > string1.Length || index < 0)

{

throw new IndexOutOfRangeException();

}

char[] mass = new char[string1.Length + string2.Length + index];

for (int i = 0; i < index; i++)

{

mass[i] = string1[i];

}

for (int i = 0; i < string2.Length; i++)

{

mass[index + i] = string2[i];

}

for (int i = index; i < string1.Length; i++)

{

mass[index + string2.Length + i] = string1[i];

}

return new string(mass);

}

/// <summary>

/// Видаляє задану кількість символів з позиції індекса

/// </summary>

/// <param name="string1">Рядок для видалення символів</param>

/// <param name="index">Індекс початку видалення</param>

/// <param name="count">Кількість символів для видалення</param>

/// <returns>Рядок з видаленими символами</returns>

/// <exception cref="IndexOutOfRangeException"></exception>

public static string Remove(string string1, int index, int count)

{

if (index > string1.Length || index < 0 || count < 0 || index + count > string1.Length)

{

throw new IndexOutOfRangeException();

}

char[] mass = new char[string1.Length - count];

for (int i = 0; i < index; i++)

{

mass[i] = string1[i];

}

for (int i = index + count; i < string1.Length; i++)

{

mass[i - count] = string1[i];

}

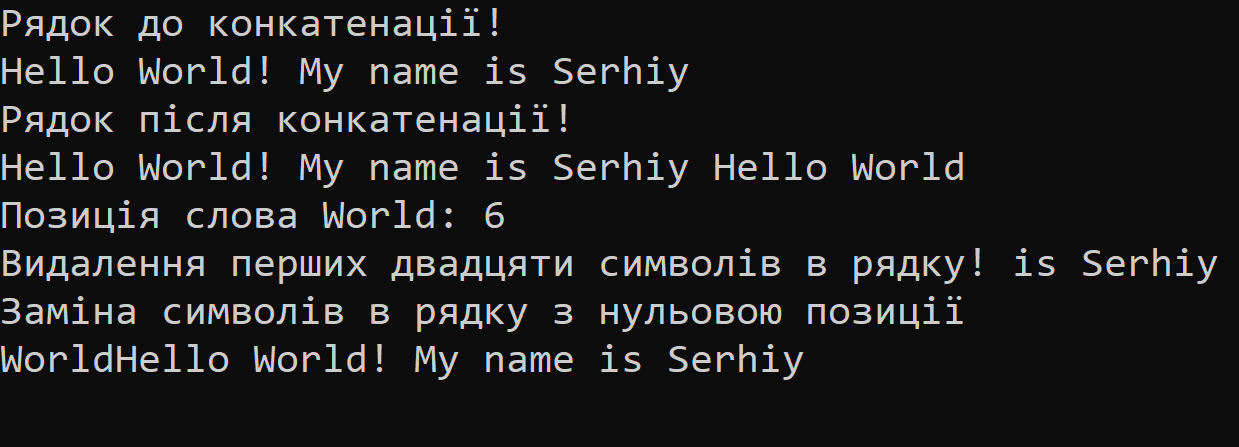
return new string(mass);

}

}

}

**Скріншот роботи бібліотеки**

****

**Аналіз відповідності реалізації розробленим специфікаціям з лабораторної роботи**

Під час написання бібліотеки всі функції були виконані згідно специфікацій лабораторної роботи №1. Відповідність реалізації функцій бібліотеки розробленим специфікаціям – стовідсоткова.

**Висновки**

Під час виконання виконання лабораторної роботи була написана бібліотека у відповідності до написаних специфікацій у першій лабораторній роботі. Для написання була вибрана мова програмування C#, вона має об’єктно-орієнтовану структуру, більш проста та зрозуміла, ніж C++, а також має деякі зручності ніж мова Java.