Міністерство освіти і науки України
Національний університет «Львівська політехніка»
Кафедра «Електронних обчислювальних машин»

Звіт
з лабораторної роботи № 1
з дисципліни: «Кросплатформенні засоби програмування»
на тему: «ДОСЛІДЖЕННЯ БАЗОВИХ КОНСТРУКЦІЙ МОВИ JAVA»

Виконав:
студент групи КІ-306
Хмільовський С. Р.
Прийняв:
доцент кафедри ЕОМ
Іванов Ю. С.

Львів – 2023

Мета роботи: ознайомитися з базовими конструкціями мови Java та оволодіти навиками написання й автоматичного документування простих консольних програм мовою Java.

Завдання (варіант № 2)

1. Написати та налагодити програму на мові Java згідно варіанту. Програма має задовольняти наступним вимогам:

- програма має розміщуватися в загальнодоступному класі Lab1ПрізвищеГрупа;
- програма має генерувати зубчатий масив, який міститиме лише заштриховані області квадратної матриці згідно варіанту;
- розмір квадратної матриці і символ-заповнювач масиву вводяться з клавіатури;
- при не введені або введенні кількох символів-заповнювачів відбувається коректне переривання роботи програми;
- сформований масив вивести на екран і у текстовий файл;
- програма має володіти коментарями, які дозволять автоматично згенерувати документацію до розробленої програми.

2. Автоматично згенерувати документацію до розробленої програми.

3. Завантажити код на GitHub згідно методичних вказівок по роботі з GitHub.

4. Скласти звіт про виконану роботу з приведенням тексту програми, результату її виконання та фрагменту згенерованої документації та завантажити його у ВНС.

5. Дати відповідь на контрольні запитання.

Вихідний код програми:

```
import java.io.*;
import java.util.*;

/**
 * Клас Lab1KhmilovskiyKI306 створює квадратну матрицю з символами-заповнювачами
та зберігає її у текстовому файлі.
 *
 * @author Khmilovskiy Stanislaw
 * @version 1.0
```

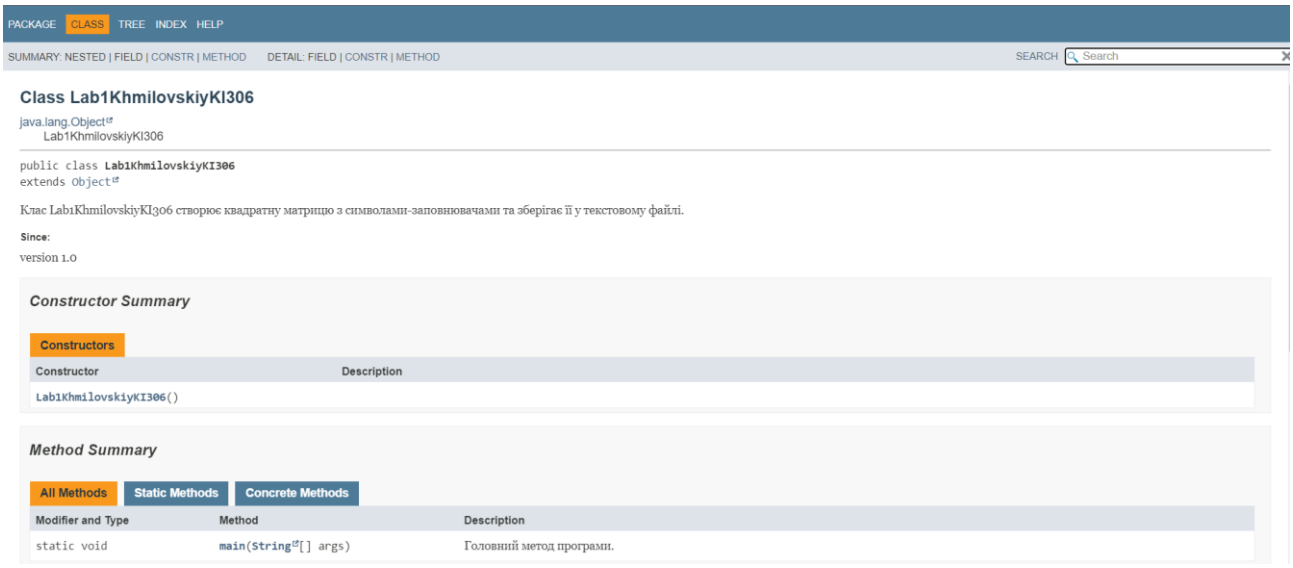
```java
 * @since version 1.0
 */

public class Lab1KhmilovskiyKI306
{
    /**
     * Головний метод програми.
     * @param args Аргументи командного рядка (не використовуються).
     * @throws FileNotFoundException Виникає, якщо файл виникає будь-яка помилка з файлом "MyFile.txt".
     */
    public static void main(String[] args) throws FileNotFoundException
    {
        int nRows;
        char[][] arr;
        String filler;
        Scanner in = new Scanner(System.in);
        File dataFile = new File("MyFile.txt");
        PrintWriter fout = new PrintWriter(dataFile);

        System.out.print("Введіть розмір квадратної матриці: ");
        nRows = in.nextInt();
        in.nextLine();
        arr = new char[nRows][];

        for (int i = 0; i < nRows; i++) {
            arr[i] = new char[i + 1];
        }

        System.out.print("\nВведіть символ-заповнювач: ");
        filler = in.nextLine();

        if (filler.length() == 1) {
            // Запис верхнього трикутника у файл і виведення
            for (int i = 0; i < nRows; i++) {
                for (int j = 0; j < i + 1; j++) {
                    arr[i][j] = filler.charAt(0);
                    System.out.print(arr[i][j]);
                    fout.print(arr[i][j]);
                }
                System.out.println();
                fout.println();
            }
            // Запис нижнього трикутника у файл і виведення
            for (int i = 1; i <= nRows; i++) {
                for (int j = 1; j <= nRows; j++) {
                    // Перехід до останнього символу "великого трикутника":
                    System.out.print(" ");
                    fout.print(" ");
                }
                for (int k = 1; k <= i; k++) {
                    arr[i - 1][k - 1] = filler.charAt(0);
                    System.out.print(arr[i - 1][k - 1]);
                    fout.print(arr[i - 1][k - 1]);
                }
                System.out.println();
                fout.println();
            }
        }
        else {
            System.out.print("\nНекоректний символ-заповнювач!");
            fout.print("\nНекоректний символ-заповнювач!");
        }

        System.out.print("\n");
        fout.print("\n");
```

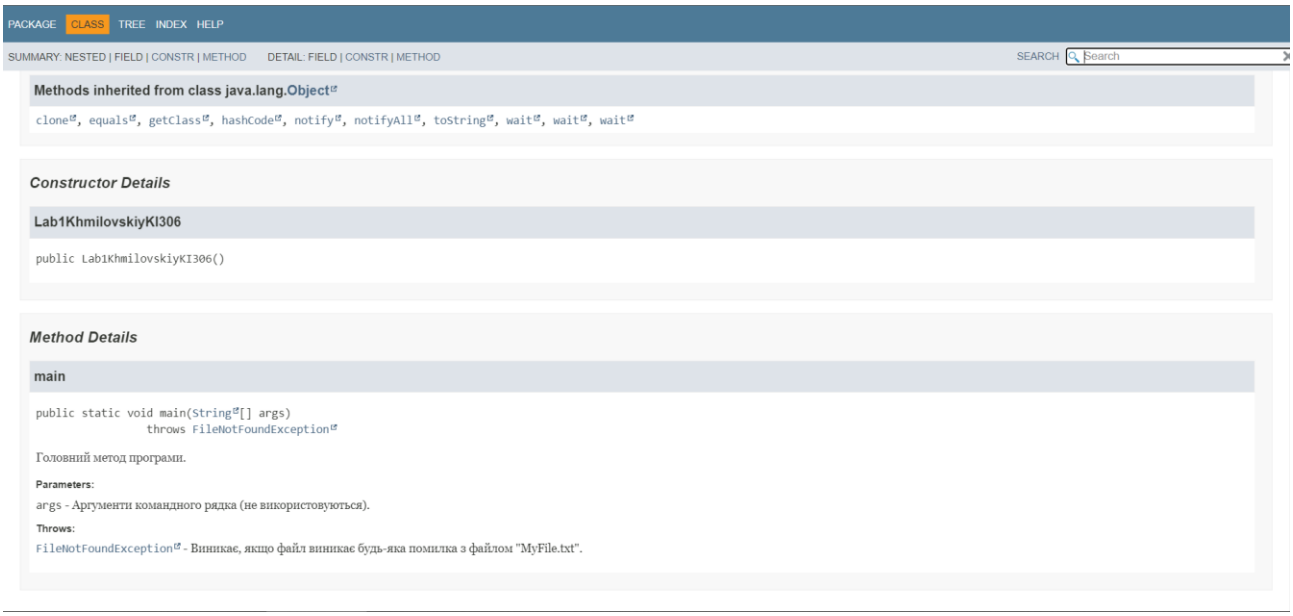
```
            fout.flush();
            fout.close();
        }
}
```

Результат виконання програми:

Вміст файлу MyFile.txt:

JavaDoc:





Відповіді на контрольні запитання:

1. Які дескриптори використовуються при коментуванні класів?

/** ... */: JavaDoc-коментарі, що дозволяють створювати докладну документацію для класу.

/* ... */: Звичайні багаторядкові коментарі для додавання загальних коментарів до класу.

// ...: Однорядкові коментарі для коротких пояснень чи коментарів до окремих рядків коду.

2. Які дескриптори використовуються при коментуванні методів?

• *@param змінна опис*

Цей дескриптор додає в опис методу розділ “parameters”. Опис цього елементу

може складатися з кількох рядків та містити html-теги. Всі дескриптори *@param*, що відносяться до одного методу слід групувати разом.

• *@return опис*

Цей дескриптор додає в опис методу розділ “returns”. Опис цього елементу може складатися з кількох рядків та містити html-теги.

• *@throws опис_класу*

Цей дескриптор додає в опис методу інформацію про класи об’єкти яких можуть генеруватися при виключних ситуаціях. Відомості про кожен клас слід описувати в окремому дескрипторі @throws.

3. Як автоматично згенерувати документацію?

Для генерування документації по пакету слід ввести в консолі ОС Windows:

javadoc –d каталог_doc ім’я_пакету

Опція *–d каталог_doc* задає каталог, де слід розмістити згенеровану документацію до пакету.

4. Які прості типи даних підтримує Java?

byte: 8-бітне ціле число.

short: 16-бітне ціле число.

int: 32-бітне ціле число.

long: 64-бітне ціле число.

float: 32-бітне число з рухомою комою (число з плаваючою точкою).

double: 64-бітне число з рухомою комою (число з плаваючою точкою).

char: 16-бітний символ Unicode.

boolean: Логічний тип, може бути true або false.

5. Як оголосити змінну-масив?

Приклади оголошення неініціалізованого одновимірного масиву типу int:

int[] arr;

int arr[];

6. Які керуючі конструкції підтримує Java?

if-else: Умовний оператор для виконання коду на основі умови.

switch: Мультиплікаційний оператор для вибору одного з багатьох можливих шляхів виконання коду на основі значення виразу.

for: Цикл для повторення коду певну кількість разів або на основі ітерації.

while: Цикл, який виконується, поки умова істинна.

do-while: Цикл, який виконується принаймні один раз, після чого перевіряється умова.

break: Використовується для виходу із циклу або вибору виразу.

continue: Використовується для переходу до наступної ітерації циклу.

return: Використовується для повернення значення з методу.

throw: Використовується для викидання виняткової ситуації.

try-catch-finally: Використовується для обробки виняткових ситуацій.

7. В чому різниця між різними варіантами оператора for?

У Java цей оператор має 2 різновиди:

• конструкція в стилі C/C++ з полем ініціалізації, логічною умовою та кроком;

• конструкція з *синтаксисом foreach* (foreach дозволяє послідовно перебирати всі елементи набору даних без застосування лічильника.).

8. Як здійснити ввід з консолі?

Для введення інформації з консолі необхідно створити об’єкт класу *Scanner* і зв’язати його з стандартним потоком вводу *System.in*, наприклад:

Scanner in = new Scanner(System.in);

9. Як здійснити ввід з текстового файлу?

Для введення інформації з файлу необхідно підключити пакет *java.io* та створити об’єкт класу *Scanner* з об’єкту *File*:

Scanner fin = new Scanner(File("MyFile.txt"));

10. Як здійснити запис у текстовий файл?

Для виведення інформації у текстовому вигляді у файл треба підключити пакет *java.io* та створити об’єкт класу *PrintWriter* в конструкторі якого необхідно вказати назву файлу, що відкривається на запис, наприклад:

PrintWriter fout = new PrintWriter ("MyFile.txt");

Висновок:

На даній лабораторній роботі я ознайомився з інтегрованим середовищем розробки, також ознайомився з базовими конструкціями мови Java та оволодів навиками написання й автоматичного документування простих консольних програм мовою Java.