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-304

Мох М. П.

**Прийняв:** Олексів М. В.

Львів – 2023

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

# Завдання (варіант № 12)

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 (рис. 1);

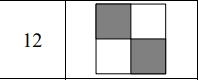


Рис. 1. Заштрихована область квадратної матриці.

* 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.

1. Автоматично згенерувати документацію до розробленої програми.
2. Скласти звіт про виконану роботу з приведенням тексту програми, результату її виконання та фрагменту згенерованої документації.
3. Дати відповіді на контрольні запитання:

* які дескриптори використовуються при коментуванні класів?
* які дескриптори використовуються при коментуванні методів?
* як автоматично згенерувати документацію?
* які прості типи даних підтримує java?
* як оголосити змінну-масив?
* які керуючі конструкції підтримує java?
* в чому різниця між різними варіантами оператора for?
* як здійснити ввід з консолі?
* як здійснити ввід з текстового файлу?
* як здійснити запис у текстовий файл?

# Вихідний код програми:

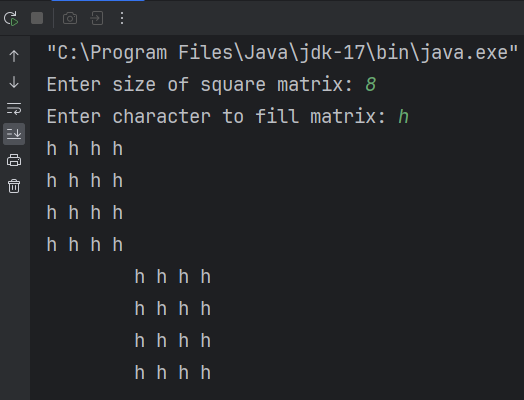
package KI304.Moh.Lab1;

import java.io.FileNotFoundException;  
import java.io.PrintWriter;  
import java.util.Scanner;  
  
*/\*\*  
 \* The Main class prompts the user for inputs, including the size of the square matrix  
 \* and a character to fill it. It then generates a jagged square matrix based on the  
 \* provided parameters, displays the matrix, and saves it to a text file.  
 \*  
 \* @author Marian Moh  
 \* @version 1.0  
 \* @since version 1.0  
 \*  
 \*/*

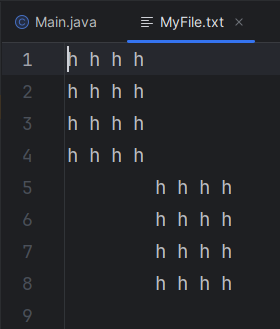
public class Lab1MohKI304 {  
  
 */\*\*  
 \* File name for saving the generated jagged matrix.  
 \*/* private static final String *FILE\_NAME* = "MyFile.txt";  
  
 */\*\*  
 \* The static main method is the entry point to the program. It calls other methods to perform  
 \* matrix generation, display, and file saving.  
 \*  
 \* @param args The command-line arguments passed to the program.  
 \*  
 \*/* public static void main(String[] args) {  
 int size = *inputSize*();  
 char filler = *inputFiller*();  
  
 char[][] juggedArr = *generateJaggedArr*(size, filler);  
  
 *showJuggedArr*(juggedArr);  
 *saveJuggedArrToFile*(juggedArr);  
 }  
  
 */\*\*  
 \* Reads and validates the size of the square matrix from the user.  
 \* It handles input exceptions and keeps prompting until a valid size is provided.  
 \*  
 \* @return int size of matrix  
 \*/* public static int inputSize() {  
 Scanner scanner = new Scanner(System.*in*);  
 int size;  
  
 while(true){  
 try {  
 System.*out*.print("Enter size of square matrix: ");  
 size = scanner.nextInt();  
 break;  
 }  
 catch (Exception e) {  
 System.*out*.println(e.getMessage() + ". Please enter a number");  
 scanner.next();  
 }  
 }  
 return size;  
 }  
  
 */\*\*  
 \* Reads and validates the character to fill the matrix from the user.  
 \* It handles input exceptions and keeps prompting until a valid character is provided.  
 \*  
 \* @return The character symbol provided by the user.  
 \*/* public static char inputFiller() {  
 Scanner scanner = new Scanner(System.*in*);  
 char filler;  
  
 while (true) {  
 System.*out*.print("Enter character to fill matrix: ");  
 String str = scanner.nextLine().trim();  
  
 if (!str.isEmpty()) {  
 filler = str.charAt(0);  
 break;  
 } else {  
 System.*out*.println("Please enter a non-empty character");  
 }  
 }  
  
 return filler;  
 }  
  
 */\*\*  
 \* Generates a jagged square matrix  
 \* of the specified size and fills it with the provided character.  
 \*  
 \* @param size The size of the square matrix to be generated.  
 \* @param filler The character to fill the matrix with.  
 \* @return The generated jagged square matrix as a 2D character array.  
 \*/* public static char[][] generateJaggedArr(int size, char filler) {  
 char[][] juggedArr = new char[size][size];  
  
 for (int i = 0; i < size; i++) {  
 for (int j = 0; j < size; j++) {  
 if(((i < size / 2) && (j < size / 2))  
 || ((i >= size / 2) && (j >= size / 2))) {  
 juggedArr[i][j] = filler;  
 } else {  
 juggedArr[i][j] = ' ';  
 }  
 }  
 }  
  
 return juggedArr;  
 }  
  
 */\*\*  
 \* Displays the generated jagged square matrix  
 \* in the console.  
 \*  
 \* @param juggedArr The jagged square matrix to be displayed.  
 \*/* public static void showJuggedArr(char[][] juggedArr) {  
 for (char[] charArr : juggedArr) {  
 for (char ch : charArr) {  
 System.*out*.print(ch + " ");  
 }  
 System.*out*.println();  
 }  
 }  
  
 */\*\*  
 \* Saves the generated jagged square matrix  
 \* to a text file named "lab2/MyFile.txt".  
 \*  
 \* @param juggedArr The jagged square matrix to be saved to the txt file.  
 \*/* public static void saveJuggedArrToFile(char[][] juggedArr) {  
 try (PrintWriter writer = new PrintWriter(*FILE\_NAME*)) {  
 for (char[] charArr : juggedArr) {  
 for (char ch : charArr) {  
 writer.print(ch + " ");  
 }  
 writer.println();  
 }  
 } catch (FileNotFoundException e) {  
 System.*out*.println(e.getMessage());  
 }  
 }  
}

**Результат виконання:**

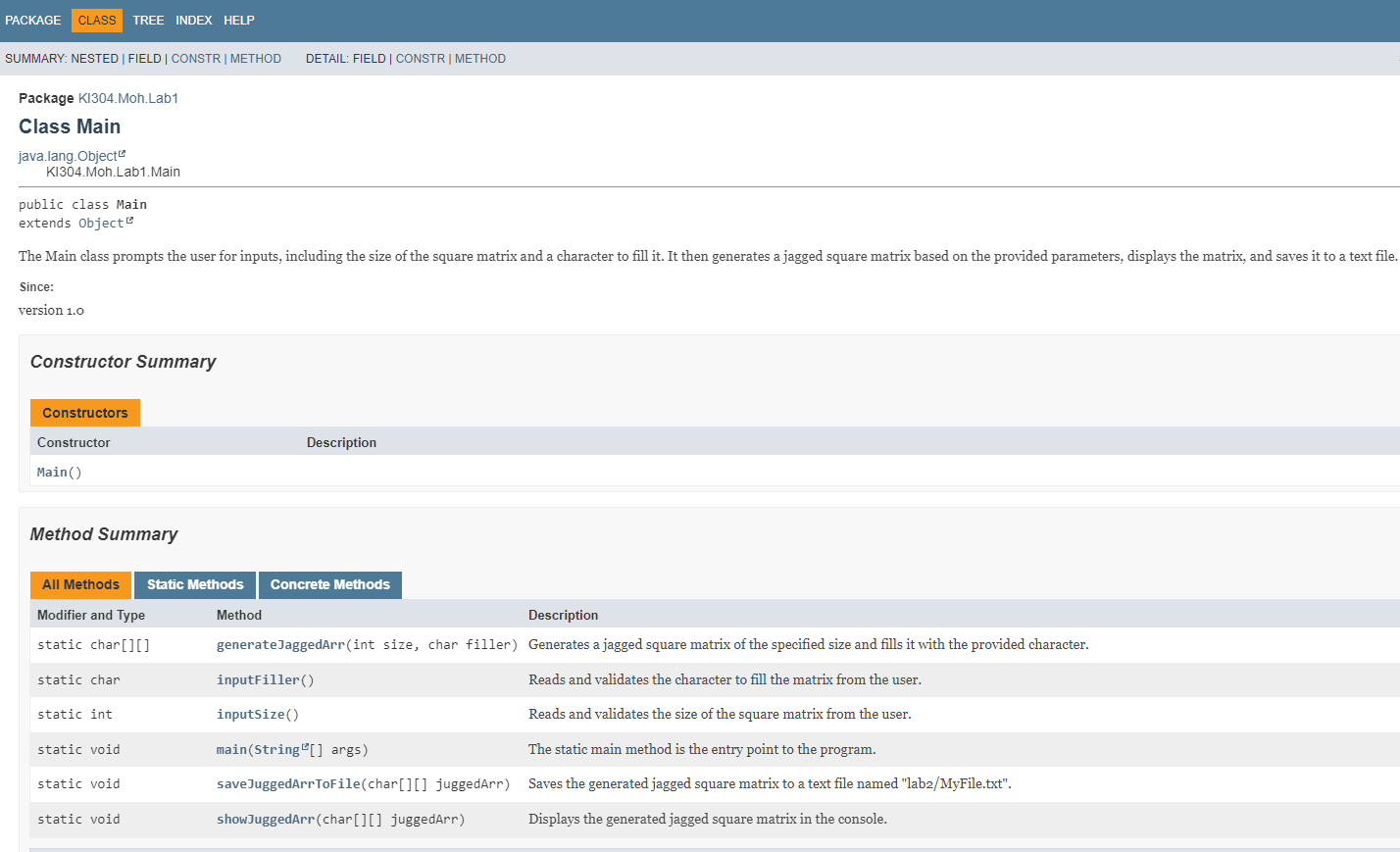
Консоль



MyFile.txt



# Фрагмент згенерованої документації:



# Відповіді на контрольні запитання:

* які дескриптори використовуються при коментуванні класів?
  + @class, @author, @version, та @since
* які дескриптори використовуються при коментуванні методів?
  + @param, @return, @throws, @deprecated, та інші, що пояснюють параметри, повернене значення, винятки, та інше.
* як автоматично згенерувати документацію?
  + за допомогою коментарів з дескрипторами та використанням інструментів, таких як JavaDoc.
* які прості типи даних підтримує java?
  + цілі числа (int, byte, short, long), числа з плаваючою комою (float, double), символи (char), та логічний тип (boolean).
* як оголосити змінну-масив?
  + тип\_даних[] ім'я\_масиву; наприклад:

int[] numbers;

* які керуючі конструкції підтримує java?
  + if, else, switch, for, while, та do-while.
* в чому різниця між різними варіантами оператора for?
  + у синтаксисі та специфіці використання. Наприклад, for зі звичайним лічильником використовується для ітерації з фіксованою кількістю ітерацій, а for-each використовується для ітерації по колекціях.
* як здійснити ввід з консолі?
  + Для ввіду з консолі використовують клас Scanner, наприклад:

Scanner scanner = new Scanner(System.in);

* як здійснити ввід з текстового файлу?
  + Для вводу з текстового файлу використовують клас FileReader або Scanner, наприклад:

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

* як здійснити запис у текстовий файл?
  + Для запису у текстовий файл можна використовувати клас PrintWriter, наприклад:

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

# Висновок:

У ході виконання даної лабораторної роботи, отримав навички роботи з мовою програмування Java. Ознайомився з базовими конструкціями мови, такими як цикли, робота з масивами, ввід та вивід даних з консолі та текстових файлів. Крім того, навчився документувати код, використовуючи коментарі та генерацію документації.