Міністерство освіти і науки України

Національний університет „Львівська політехніка”

Кафедра “Електронних обчислювальних машин”



**з дисципліни**

**Кросплатформні засоби програмування**

**Звіт з Лабораторної роботи №1**

**ДОСЛІДЖЕННЯ БАЗОВИХ КОНСТРУКЦІЙ МОВИ JAVA**

**Виконав:**

КІ-307

Герега Р. О.

**Перевірив:**

Іванов Ю.С

**2023**

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

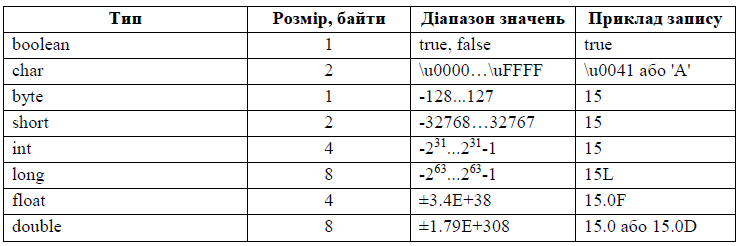
**Теоретичний вступ**

**Автоматичне документування**

При автоматичній генерації документації використовується утиліта javadoc, яка аналізує вміст між /\*\* і \*/ та на його базі генерує документацію у форматі \*.html. Коментарі між /\*\* і \*/ прийнято починати з описового тексту, за яким слідують дескриптори. Використання дескрипторів полегшує як автоматичну генерацію документації, так і розуміння коду, до якого відноситься коментар. Дескриптор, на відміну від решти коментарів, починається з символу @ за яким слідує ім’я дескриптора. Оскільки документація генерується у форматі \*.html, то між /\*\* і \*/ допускається розташування html-тегів, включаючи рисунки.

**Основні типи мови Java**

Мова Java є строго типізованою. Це означає, що тип кожної змінної має бути оголошеним. Мова має 8 основних (простих) типів, які не є класами та однаково представляються на будь-якій машині, де виконується програма.



**Масиви**

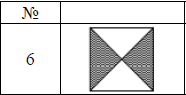
Масив – структура даних, що зберігає набір значень однакового типу. Пам’ять під масив виділяється у керованій кучі. При завершенні життєвого циклу масиву пам’ять, яку він займав, вивільняється збирачем сміття. Доступ до елементів масиву здійснюється за допомогою індексів. Індексація масивів у Java починається з 0. Для створення масиву у Java необхідно оголосити змінну-масив та ініціалізувати її. При створенні за допомогою оператора new масиву чисел всі його елементи ініціалізуються нулями (масиви типу boolean ініціалізуються значеннями false, масиви об’єктів ініціалізуються значеннями null). Після

створення масиву змінити його розмір неможливо.

**Зубчаті масиви**

Завдяки тому, що багатовимірні масиви у Java реалізуються як множина одновимірних масивів, стає можливим реалізувати багатовимірні масиви з різною кількістю елементів у межах виміру. Синтаксис оголошення зубчатого масиву нічим не відрізняється від синтаксису оголошення звичайного багатовимірного масиву. Різниця є лише у способі ініціалізації, де використовується виділення пам’яті під різну кількість елементів у межах виміру.

**Варіант 6**



**Завдання:**

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.

**Код**

package lab1;

import java.io.\*;

import java.util.Scanner;

/\*\*

\* Клас Lab1HerehaKI307 реалізує програму згідно варіанту 6 до лабораторної роботи №1

\*/

public class Lab1HerehaKI307 {

/\*\*

\* Статичний метод main є точкою входу в програму

\*

\*@author Hereha Rostyslav

\* @param args

\* @throws FileNotFoundException

\*

\*/

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.println("Введіть розмір квадратної матриці: ");

int matrixSize = scanner.nextInt();

// Перевірка, чи введено коректний розмір матриці

if (matrixSize <= 0) {

System.out.println("Некоректний розмір матриці. Програма завершує роботу.");

return;

}

System.out.println("Введіть символ-заповнювач: ");

char fillCharacter = scanner.next().charAt(0);

int fillCounter = 0;

int step = 1;

// Створення та заповнення зубчатого масиву

char[][] jaggedArray = new char[matrixSize][];

for (int i = 0; i < matrixSize; i++) {

jaggedArray[i] = new char[matrixSize];

for (int j = 0; j < matrixSize; j++) {

if(j <= fillCounter) {

jaggedArray[i][j] = fillCharacter;

}

if(j >= matrixSize - fillCounter - 1) {

System.out.print("j: " + j + " i: " + i + "\n");

jaggedArray[i][j] = fillCharacter;

}

if (fillCounter == matrixSize/2) {

step = -1;

}

}

fillCounter = fillCounter + step;

}

// Виведення масиву на екран

System.out.println("Згенерований зубчатий масив:");

for (int i = 0; i < matrixSize; i++) {

for (int j = 0; j < matrixSize; j++) {

System.out.print(jaggedArray[i][j] + " ");

}

System.out.println();

}

// Запис масиву у текстовий файл

try (PrintWriter writer = new PrintWriter("output.txt")) {

for (int i = 0; i < matrixSize; i++) {

for (int j = 0; j < matrixSize; j++) {

writer.print(jaggedArray[i][j] + " ");

}

writer.println();

}

System.out.println("Масив був записаний у файл 'output.txt'.");

} catch (IOException e) {

System.out.println("Помилка при записі у файл.");

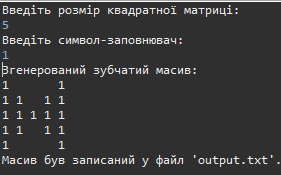
e.printStackTrace();

}

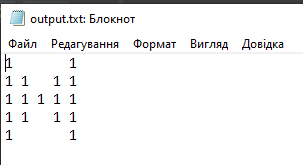
}

}

**Результат:**



**Output.txt:**



**Короткий висновок**

Під час виконання даної лабораторної роботи я ознайомився з базовими конструкціями мови Java та оволодіти навиками написання й автоматичного документування простих консольних програм мовою Java.