Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Кафедра ЕОМ



**Звіт**

до лабораторної роботи № 1

з дисципліни: «Кросплатформні засоби програмування»

на тему: «**ДОСЛІДЖЕННЯ БАЗОВИХ КОНСТРУКЦІЙ МОВИ JAVA**»

Виконав:

ст.гр. КІ-307

Магора О.Т.

Прийняв:

Іванов Ю.С.

**Львів 2023**

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

**ТЕОРЕТИЧНІ ВІДОМОСТІ**

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.

Для автоматичної генерації документації між /\*\* і \*/ можна розмістити:

• коментарі до класу;

• коментарі до методів;

• коментарі до полів;

• загальні коментарі.

Коментарі до класу мають бути розміщені після директив import безпосередньо перед визначенням класу. Найчастіше цей коментар має вигляд одного або кількох коротких речень:

/\*\*

Об’єкт класу Person описує особу.

Особа має властивості: ім’я, прізвище та стать.

\*/

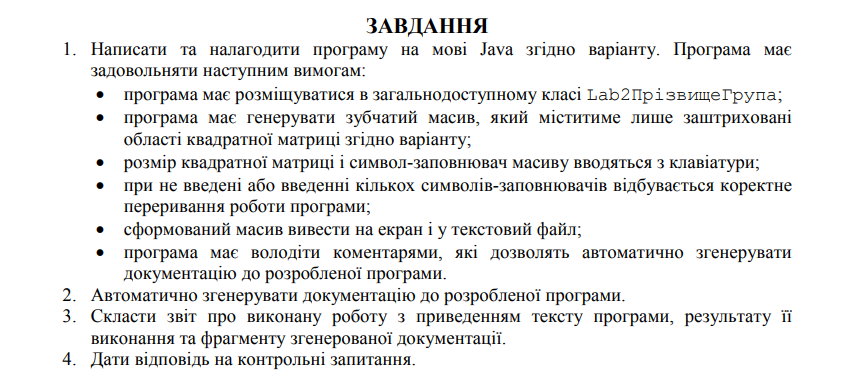
Коментарі до методів розташовуються безпосередньо перед методами, які вони описують. Крім дескрипторів загального призначення для коментування методів використовуються дескриптори:

• @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.

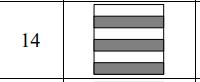
• @return опис Цей дескриптор додає в опис методу розділ “returns”. Опис цього елементу може складатися з кількох рядків та містити html-теги.

• @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.



**Варіант:**



**Код:**

package lab1;

import java.io.\*;

import java.util.\*;

public class Lab1\_Magora\_KI\_307 {

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();

System.out.print("\nВведіть символ-заповнювач: ");

filler = in.nextLine();

if (filler.length() != 1) {

System.out.print("\nНе коректний символ заповнювач");

fout.flush();

fout.close();

return;

}

int correctNRows = (int) Math.floor(nRows / 2);

arr = new char[correctNRows][];

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

arr[i]= new char[nRows];

}

char normalizedFiller = (char) filler.codePointAt(0);

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

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

fout.print("\n");

for(int j = 0; j < arr[i].length; j++) {

arr[i][j] = normalizedFiller;

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

fout.print(arr[i][j] + " ");

}

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

fout.print("\n");

}

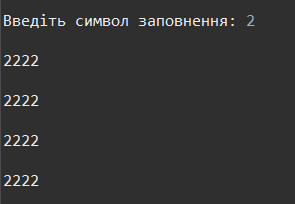
fout.flush();

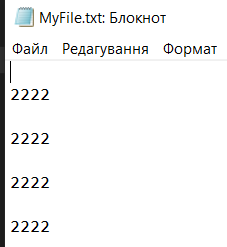
fout.close();

}

}

**Виконання програми:**





**Висновок:** Ознайомився з базовими конструкціями мови Java та оволодів навиками написання й автоматичного документування простих консольних програм мовою Java.