**Міністерство освіти і науки України**

**Національний університет «Львівська політехніка»**

**Кафедра ЕОМ**



Звіт

до лабораторної роботи № 1

з дисципліни: «Кросплатформні засоби програмування»

«Дослідження базових конструкцій мови Java»

Варіант №9

Виконав:

ст.гр. КІ-34

Корнєєв А.В.

Прийняв:

Іванов Ю.С.

**Львів 2022**

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

**Завдання:**

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

• програма має розміщуватися в загальнодоступному класі Lab2ПрізвищеГрупа;

• програма має генерувати зубчатий масив, який міститиме лише заштриховані області квадратної матриці згідно варіанту;

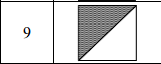
• розмір квадратної матриці і символ-заповнювач масиву вводяться з клавіатури;

• 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.

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 відповідь на контрольні запитання.



**Теоретичний матеріал:**

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. 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.

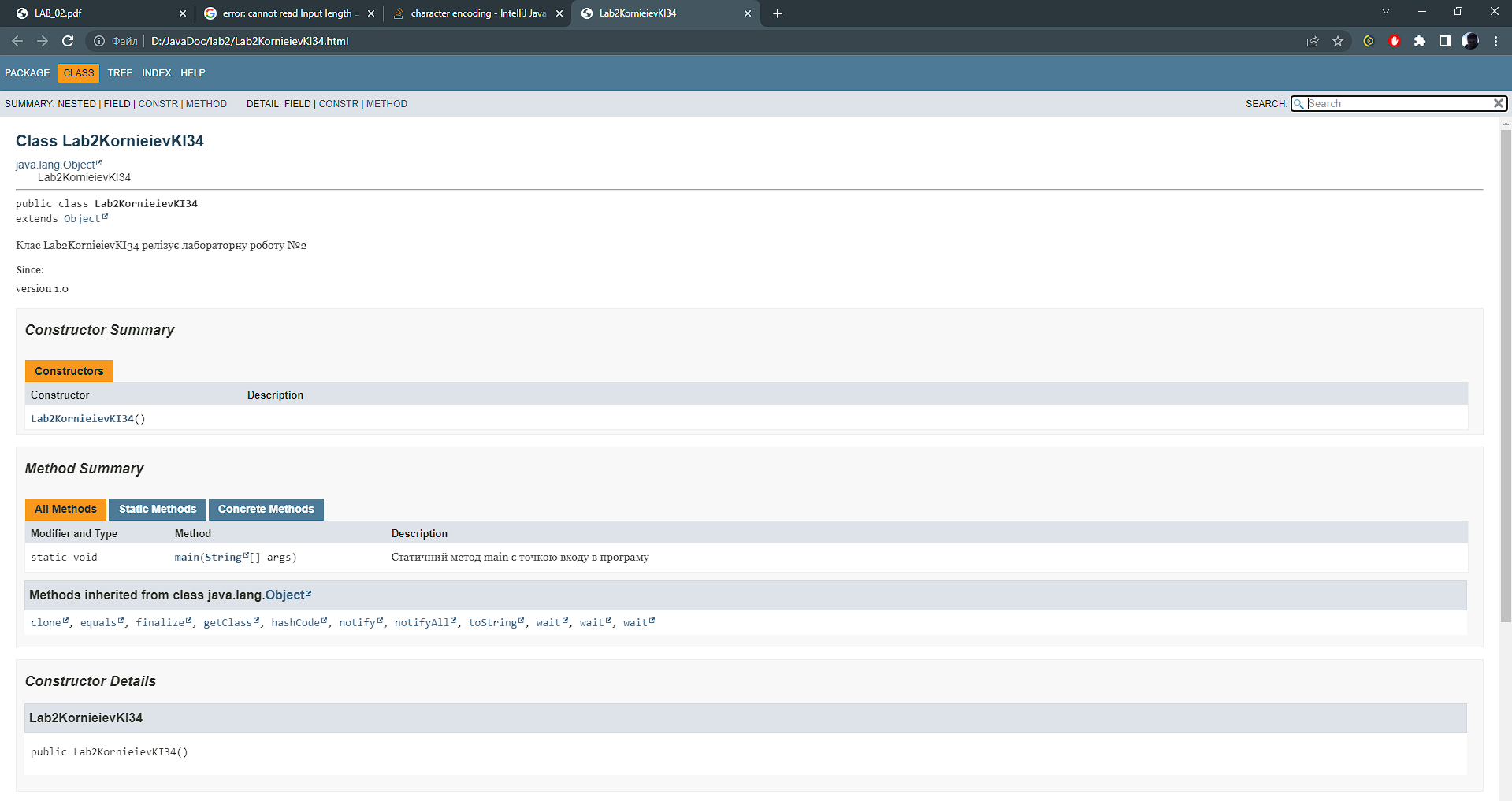
• @return опис Цей дескриптор додає в опис методу розділ “returns”. Опис цього елементу може складатися з кількох рядків та містити html-теги.

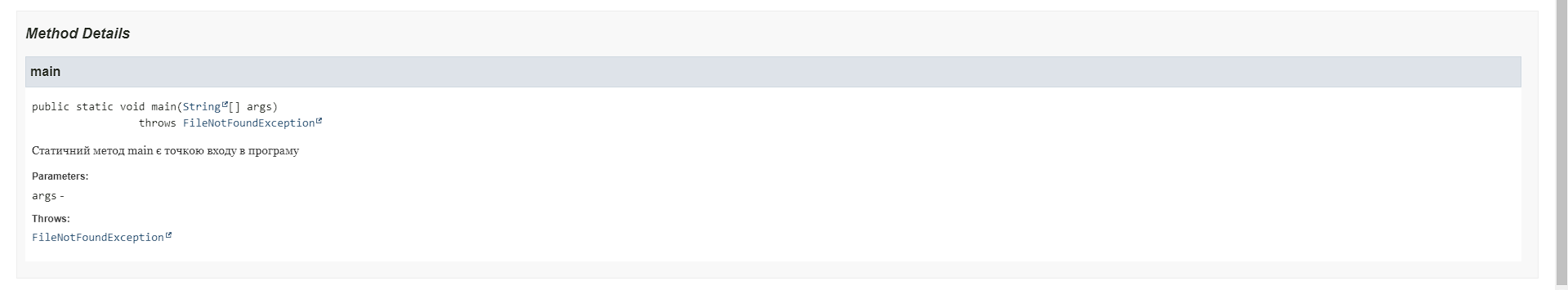
• @throws опис\_класу

**Код програми :**

import java.io.File;  
import java.io.FileNotFoundException;  
import java.util.Scanner;  
import java.util.Arrays;  
import java.io.PrintWriter;  
  
*/\*\*  
 \* Клас Lab2KornieievKI34 релізує лабораторну роботу №2  
 \*  
 \** ***@author*** *chemp  
 \** ***@version*** *1.0  
 \** ***@since*** *version 1.0  
 \*  
 \*/*public class Lab2KornieievKI34  
{  
  
 */\*\*  
 \* Статичний метод main є точкою входу в програму  
 \*  
 \** ***@param*** *args  
 \** ***@throws*** *FileNotFoundException  
 \*  
 \*/* 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();  
  
  
 exit:  
 for(int i = nRows-1; i >= 0; i--)  
 {  
 for(int j = 0; j < i+1; j++)  
 {  
 if(filler.length() == 1)  
 {  
  
 arr[i][j] = (char) filler.codePointAt(0);  
 System.*out*.print(arr[i][j] + " ");  
 fout.print(arr[i][j] + " ");  
 }  
 else if (filler.length() == 0)  
 {  
 System.*out*.print("\nНе введено символ заповнювач");  
 break exit;  
 }  
 else  
 {  
 System.*out*.print("\nЗабагато символів заповнювачів");  
 break exit;  
 }  
 }  
 System.*out*.print("\n");  
 fout.print("\n");  
 }  
 fout.flush();  
 fout.close();  
  
 }  
}

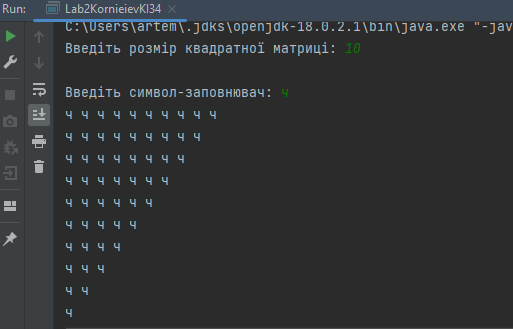
Документація



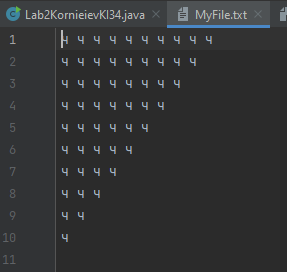


**Результат виконання програми:**

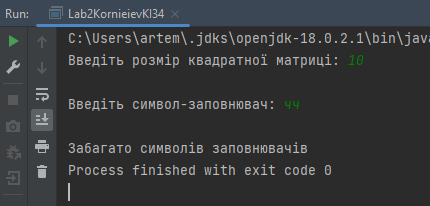
Вивід в консоль



Вивід в файл



Робота програми при некоректному вводі



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