Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Кафедра ЕОМ

Звіт

до лабораторної роботи № 5

З дисципліни: «Кросплатформенні засоби програмування»

На тему: **«ФАЙЛИ»**

Варіант 16

Виконав: ст.гр. КІ-301

Онисько М.М.

Прийняв: доц.

Майдан М.В.

Львів 2023

Мета: оволодіти навиками використання засобів мови Java для роботи з потоками і файлами.

Завдання:
1. Створити клас, що реалізує методи читання/запису у текстовому і двійковому форматах результатів роботи класу, що розроблений у лабораторній роботі №5. Написати програму для тестування коректності роботи розробленого класу.
2. Для розробленої програми згенерувати документацію.
3. Скласти звіт про виконану роботу з приведенням тексту програми, результату її виконання та фрагмент згенерованої документації.
4. Дати відповідь на контрольні запитання.

Виконання роботи:

Код програми:

Lab5Onysko.java

```java
package KI301.Onysko.Lab5;
import java.io.*;
import java.util.Scanner;

/**
 * This program is a class driver that tests the operation of the FinApp class, and
also writes data to files
 * @author Onysko Mykola KI-301
 * @version 1.0
 * @since version 1.0
*/

public class Lab5Onysko{
    public static void main(String[] args)throws IOException {
        double x;
        String fileNameTxt = "resText.txt";
        String fileNameBin = "resBin.bin";
        FinApp calco = new FinApp();

        Scanner s = new Scanner(System.in);
        System.out.print("Enter data: ");
        x = s.nextDouble();
        calco.calculate(x);

        System.out.print("Result is: "+calco.getResult()+"\n");
        calco.writeResTxt(fileNameTxt);
        calco.writeResBin(fileNameBin);

        calco.readResTxt(fileNameTxt);
        System.out.print("Result of text file: "+calco.getResult()+"\n");
        calco.readResBin(fileNameBin);
        System.out.print("Result of bin file: "+calco.getResult()+"\n");

    }
}
```

FinApp.java

```java
/**
 *
 * @author Mykola Onysko
 * @version 1.0
 */
package KI301.Onysko.Lab5;

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

/**
 * The FinApp class represents a financial calculator.
 */
public class FinApp {
    private double res;

    /**
     * Calculates the result based on the given mathematical expression.
     *
     * @param x The input value for the expression.
     * @throws ArithmeticException If the mathematical expression results in a
division by zero.
     */
    public void calculate(double x) throws ArithmeticException {
        if (Math.tan(x) == 0) {
            throw new ArithmeticException("Exception: Tan is equal to 0!!!\n");
        } else {
            res = 7 * x / Math.tan(2 * x - 4);
        }
    }

    /**
     * Writes the result to a text file.
     *
     * @param fileName The name of the text file to write the result to.
     * @throws FileNotFoundException If the specified file is not found.
     */
    public void writeResTxt(String fileName) throws FileNotFoundException {
        PrintWriter f = new PrintWriter(fileName);
        f.printf("%f ", res);
        f.close();
    }

    /**
     * Reads the result from a text file.
     *
     * @param fileName The name of the text file to read the result from.
     */
    public void readResTxt(String fileName) {
        try {
            File f = new File(fileName);
            if (f.exists()) {
                Scanner s = new Scanner(f);
                res = s.nextDouble();
                s.close();
            } else
                throw new FileNotFoundException("File " + fileName + " not found");
        } catch (FileNotFoundException ex) {
            System.out.print(ex.getMessage());
        }
    }
```

```java
    /**
     * Writes the result to a binary file.
     *
     * @param fileName The name of the binary file to write the result to.
     * @throws FileNotFoundException If the specified file is not found.
     * @throws IOException           If an I/O error occurs while writing to the
file.
     */
    public void writeResBin(String fileName) throws FileNotFoundException,
IOException {
        DataOutputStream f = new DataOutputStream(new FileOutputStream(fileName));
        f.writeDouble(res);
        f.close();
    }

    /**
     * Reads the result from a binary file.
     *
     * @param fileName The name of the binary file to read the result from.
     * @throws FileNotFoundException If the specified file is not found.
     * @throws IOException           If an I/O error occurs while reading from the
file.
     */
    public void readResBin(String fileName) throws FileNotFoundException, IOException
{
        DataInputStream f = new DataInputStream(new FileInputStream(fileName));
        res = f.readDouble();
        f.close();
    }

    /**
     * Gets the calculated result.
     * @return The calculated result.
     */
    public double getResult() {
        return this.res;
    }
}
```

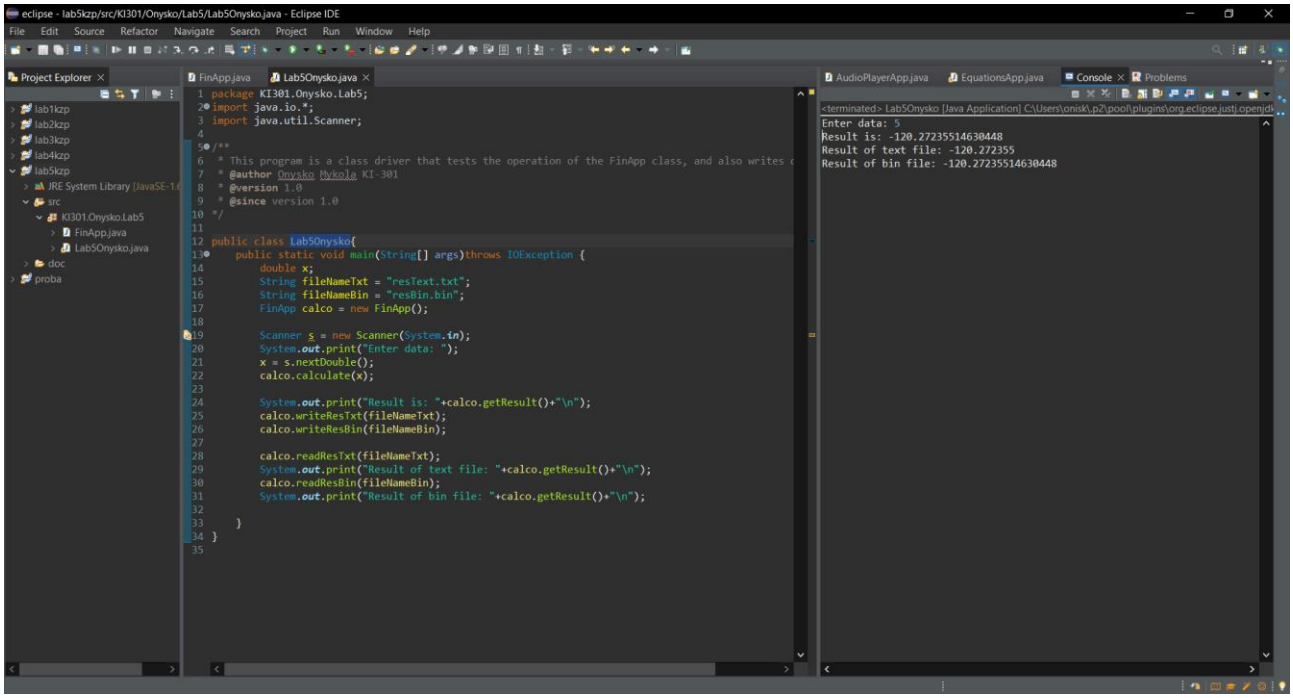

Рис.1 Код та вивід програми

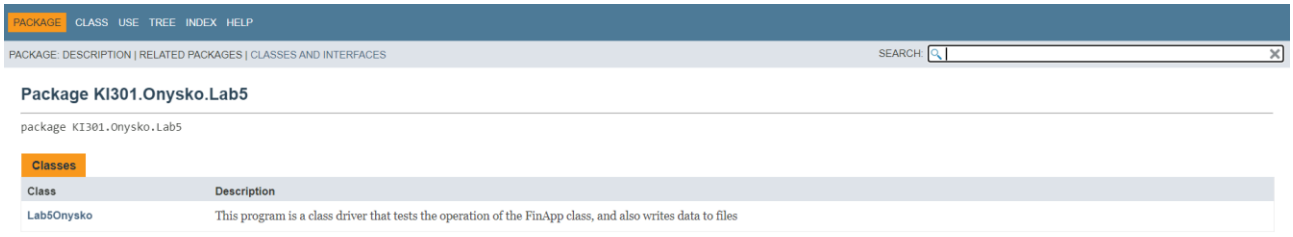

Рис.2 Документація Package KI301.Onysko.Lab5

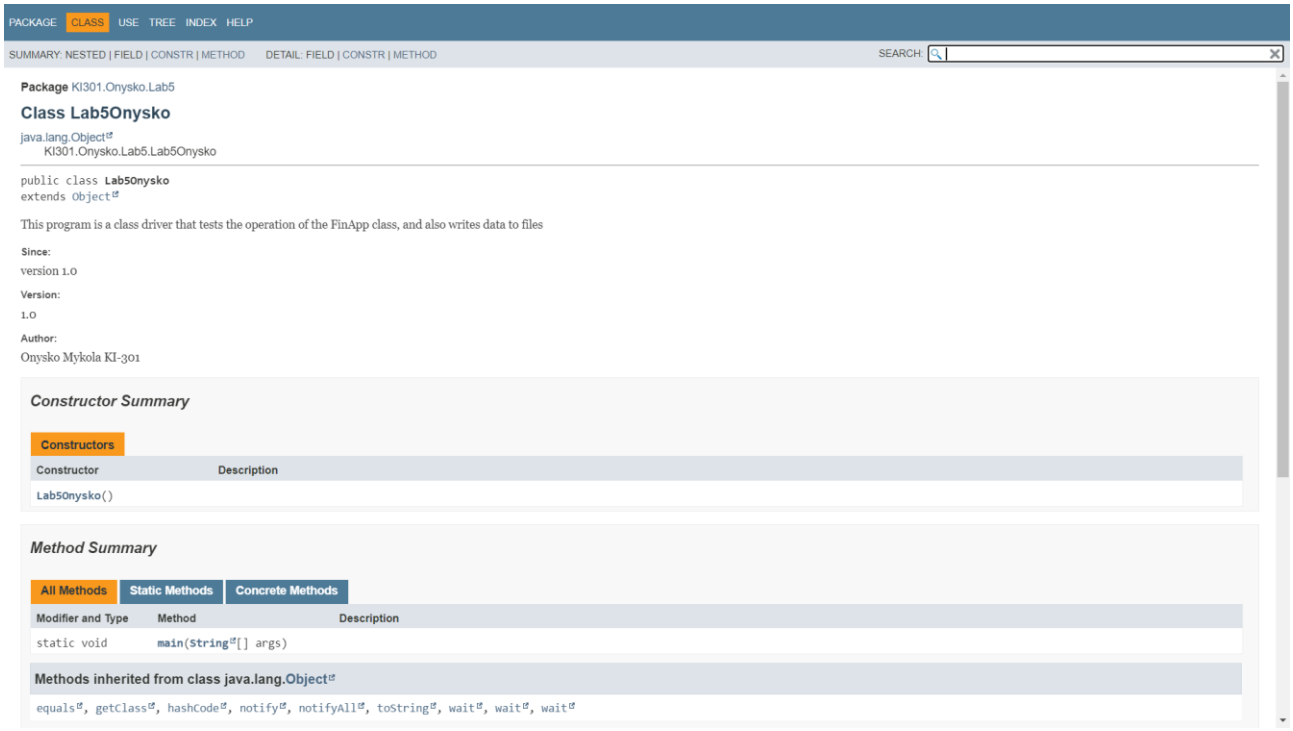

Рис.3 Документація Lab5Onysko

Висновок:

В результаті виконання цього завдання були освоєні навички роботи з потоками та файлами в мові Java. Було реалізовано ефективні механізми обробки виключень, а також створено програму-драйвер для тестування класу, який забезпечує коректну обробку даних та зберігання результатів у файл. Код був організований у пакет з автоматичною генерацією документації за допомогою Javadoc.

Контрольні питання:

1. Розкрийте принципи роботи з файловою системою засобами мови Java.

Для роботи з файловою системою в Java використовуються класи java.io.File для представлення файлів і каталогів, а також класи потоків вводу/виводу для читання та запису даних.

2. Охарактеризуйте клас Scanner.

Клас Scanner дозволяє зчитувати рядки тексту, розбиваючи їх на токени за заданим розділювачем.

3. Наведіть приклад використання класу Scanner.

Scanner scanner = new Scanner(System.in);

String line = scanner.nextLine();

4. За допомогою якого класу можна здійснити запис у текстовий потік?

Для запису в текстовий потік використовується клас PrintWriter.

5. Охарактеризуйте клас PrintWriter.

Клас PrintWriter дозволяє зручно записувати дані у символьний потік виводу.

6. Розкрийте методи читання/запису двійкових даних засобами мови Java.

Для читання/запису двійкових даних використовуються класи DataInputStream і DataOutputStream.

7. Призначення класів DataInputStream і DataOutputStream.

DataInputStream / DataOutputStream - це потоки для читання/запису даних у двійковому форматі.

8. Який клас мови Java використовується для здійснення довільного доступу до файлів.

Для довільного доступу до файлів використовується клас RandomAccessFile.

9. Охарактеризуйте клас RandomAccessFile.

Клас RandomAccessFile дозволяє читати з будь-якого місця файлу і записувати в будь-яке місце.

10. Який зв’язок між інтерфейсом DataOutput і класом DataOutputStream?

Інтерфейс DataOutput визначає методи запису даних. Клас DataOutputStream реалізує ці методи для запису даних у двійковому форматі.