Міністерство освіти і науки України Національний університет «Львівська політехніка»

Кафедра ЕОМ



Звіт

до лабораторної роботи № 5

з дисципліни: «Кросплатформні засоби програмування»

на тему: «Файли у Java»

Варіант №13

Виконав: ст. гр. KI-203 Панченко Д. В.

Прийняв: доцент кафедри ЕОМ Іванов Ю. С.

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 №4. Написати програму для тестування коректності роботи розробленого класу.
- 2. Для розробленої програми згенерувати документацію.
- 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.

Варіант завдання: $13 - y = \sin(x)/\cot(8x)$

Вихідний код програми:

Файл Equations App. java:

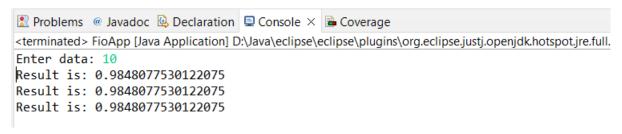
```
package KI 306.Panchenko.Lab5;
import java.util.Scanner;
import java.io.*;
import static java.lang.System.out;
* Class <code>EquationsApp</code> Implements driver for Equations class
* @author Panchenko
public class EquationsApp {
       * @param args is arguments
      public static void main(String[] args) {
             try
             {
                   out.print("Enter file name: ");
                   Scanner in = new Scanner(System.in);
                   String fName = in.nextLine();
                   PrintWriter fout = new PrintWriter(new File(fName));
                   try
                   {
                          try
                          {
                                Equations eq = new Equations();
                                out.print("Enter X: ");
                                fout.print(eq.calculate(in.nextInt()));
                          finally
                                // Цей блок виконається за будь-яких обставин
```

```
fout.flush();
                                  fout.close();
                           }
                    }
                    catch (CalcException ex)
                           // Блок перехоплює помилки обчислень виразу
                           out.print(ex.getMessage());
                    }
             }
             catch (FileNotFoundException ex)
                    // <u>Блок перехоплює помилки роботи</u> з файлом навіть якщо вони виникли
у блоці finally
                    out.print("Exception reason: Perhaps wrong file path");
             }
       }
}
* Class <code>CalcException</code> more precises ArithmeticException
* @author Panchenko
class CalcException extends ArithmeticException
       public CalcException(){}
      public CalcException(String cause)
       {
             super(cause);
       }
}
* Class <code>Equations</code> implements method for sin(x)/ctg(8x) expression
* calculation
* @author Panchenko
class Equations
{
* Method calculates the sin(x)/\underline{ctg}(8x) = sin(x) * \underline{tg}(8x) expression
* @param x Angle in degrees
* @throws CalcException is exception for calculate
* @return calculation result
      public static double calculate(double x) throws CalcException
             double y, rad;
             rad = x * Math.PI / 180.0;
             try
             {
                    y = Math.sin(rad) * Math.tan(rad*8);
                    // Якщо результат не є числом, то генеруємо виключення
                    if (y==Double.NaN || y==Double.NEGATIVE_INFINITY ||
y==Double.POSITIVE_INFINITY | | x==90 | | x== -90) {
                           throw new ArithmeticException();
                           }
             }
             catch (ArithmeticException ex)
```

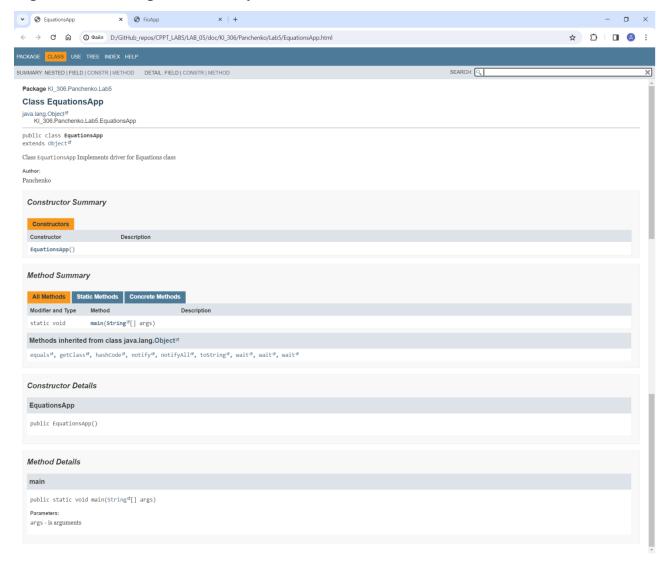
```
// <u>створимо виключення вищого рівня</u> з <u>поясненням причини виникнення</u>
помилки
                    if (rad==Math.PI/2.0 | rad==-Math.PI/2.0)
                           {throw new CalcException("Exception reason: Illegal value of
X for cotangent calculation");}
                    else
                           {throw new CalcException("Unknown reason of the exception
during exception calculation");}
             }
             return y;
      }
}
Файл FioApp.java:
package KI 306.Panchenko.Lab5;
import java.io.*;
import java.util.*;
* Class <code>FioApp</code> Inheritance class for Equations class
* @author Panchenko
public class FioApp extends Equations{
       * @param args is arguments
       * # @throws FileNotFoundException is exception
       */
      public static void main(String[] args) throws FileNotFoundException, IOException
             // TODO Auto-generated method stub
             CalcWFio obj = new CalcWFio();
             Scanner \underline{s} = new Scanner(System.in);
             System.out.print("Enter data: ");
             double data = s.nextDouble();
             obj.calc(data);
             System.out.println("Result is: " + obj.getResult());
             obj.writeResTxt("textRes.txt");
             obj.writeResBin("BinRes.bin");
             obj.readResBin("BinRes.bin");
             System.out.println("Result is: " + obj.getResult());
             obj.readResTxt("textRes.txt");
             System.out.println("Result is: " + obj.getResult());
      }
}
* Class <code>CalcWFio</code> Implementation of functions for the class FioApp
* @author Panchenko
class CalcWFio
{
```

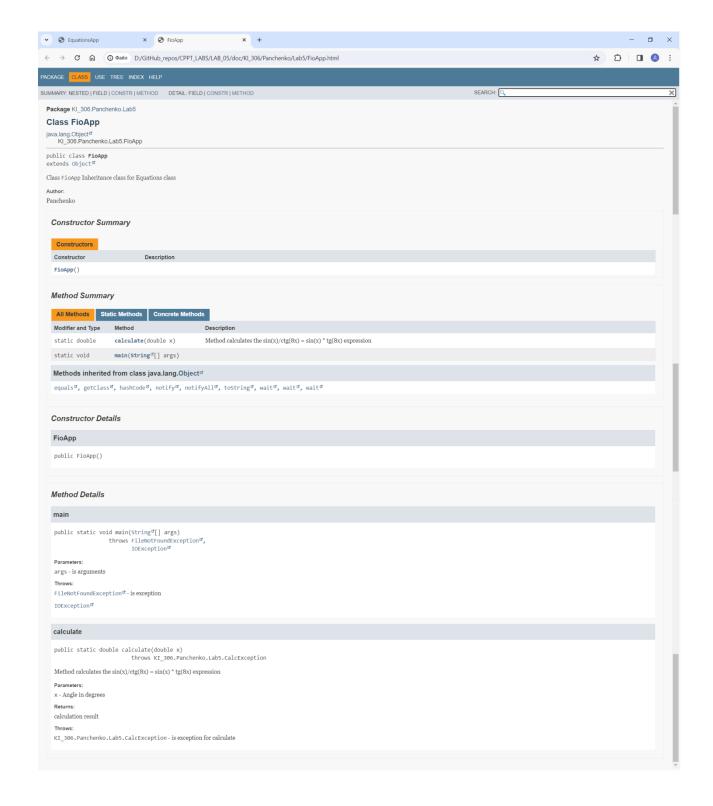
```
public void writeResTxt(String fName) throws FileNotFoundException
      {
             PrintWriter f = new PrintWriter(fName);
             f.printf("%.17f ",result);
             f.close();
      }
      public void readResTxt(String fName)
             try
             {
                   File f = new File (fName);
                          if (f.exists())
                          {
                                 Scanner s = new Scanner(f);
                                 result = s.nextDouble();
                                 s.close();
                          }
                          else
                                 throw new FileNotFoundException("File " + fName + "not
found");
             catch (FileNotFoundException ex)
             {
                   System.out.print(ex.getMessage());
      }
      public void writeResBin(String fName) throws FileNotFoundException, IOException
             DataOutputStream f = new DataOutputStream(new FileOutputStream(fName));
             f.writeDouble(result);
             f.close();
      public void readResBin(String fName) throws FileNotFoundException, IOException
             DataInputStream f = new DataInputStream(new FileInputStream(fName));
             result = f.readDouble();
             f.close();
      }
      public void calc(double x)
             result = Equations.calculate(x);
      }
      public double getResult()
      {
             return result;
      private double result;
}
```

Результат виконання програми:



Фрагмент згенерованої документації:





Відповіді на контрольні запитання:

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

Для створення файлових потоків і роботи з ними у Java ϵ 2 класи, що успадковані від InputStream і OutputStream це - FileInputStream і FileOutputStream. Як і їх суперкласи вони мають методи лише для байтового небуферизованого блокуючого читання/запису даних та керуванням потоками.

2. Охарактеризуйте клас Scanner.

Для читання текстових потоків найкраще підходить клас Scanner. На відміну від InputStreamReader і FileReader, що дозволяють лише читати текст, він має велику кількість методів, які здатні читати як рядки, так і окремі примітивні типи з подальшим їх перекодуванням до цих типів, робити шаблонний аналіз текстового потоку, здатний працювати без потоку даних та ще багато іншого.

3. Наведіть приклад використання класу Scanner.

```
Приклад читання даних за допомогою класу Scanner з стандартного потоку вводу: Scanner sc = new Scanner(System.in); int i = sc.nextInt(); Приклад читання даних за допомогою класу Scanner з текстового файлу: Scanner sc = new Scanner(new File("myNumbers")); while (sc.hasNextLong()) { long aLong = sc.nextLong(); }
```

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

Для буферизованого запису у текстовий потік найкраще використовувати клас PrintWriter.

5. Охарактеризуйте клас PrintWriter.

Цей клас має методи для виводу рядків і чисел у текстовому форматі: print, println, println, - принцип роботи яких співпадає з аналогічними методами Systen.out.

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

Читання двійкових даних примітивних типів з потоків здійснюється за допомогою класів, що реалізують інтерфейс DataInput. Інтерфейс DataInput визначає такі методи для читання двійкових даних:

- readByte;
- readInt;
- readShort;
- readLong;
- readFloat:
- readDouble:
- readChar;
- readBoolean;
- readUTF.

Запис двійкових даних примітивних типів у потоки здійснюється за допомогою класів, що реалізують інтерфейс DataOutput. Інтерфейс DataOutput визначає такі методи для запису двійкових даних:

- writeByte;
- writeInt;
- writeShort;
- writeLong;
- writeFloat;
- writeDouble:
- writeChar;
- writeChars;
- writeBoolean:
- writeUTF.

7. Призначення класів DataInputStream і DataOutputStream.

DataInputStream в Java використовується для зчитування примітивних типів даних та рядків з бінарних потоків вводу, тоді як DataOutputStream використовується для запису цих типів у бінарний потік виводу.

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

Керування файлами з можливістю довільного доступу до них здійснюється за допомогою класу RandomAccessFile.

9. Охарактеризуйте клас RandomAccessFile.

Відкривання файлу в режимі запису і читання/запису здійснюється за допомогою конструктора, що приймає 2 параметри – посилання на файл (File file) або його адресу (String name) та режим відкривання файлу (String mode):

RandomAccessFile(File file, String mode); RandomAccessFile(String name, String mode).

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

Інтерфейс DataOutput визначає методи для запису примітивних типів та рядків у байтовий потік, тоді як клас DataOutputStream унаслідований від FilterOutputStream і реалізує цей інтерфейс, надаючи конкретну реалізацію для запису даних у вихідний потік байтів.

Висновок: Я оволодів навиками використання засобів мови Java для роботи з потоками і файлами.