Міністерство освіти і науки України

Національний університет “Львівська політехніка”

Кафедра ЕОМ

Звіт

з лабораторної роботи №6

з дисципліни: «**Кросплатформенні засоби програмування**»

на тему: «**Файли**»

Виконав: ст.гр. КІ-34

Скалій Т.В.

Прийняв:

викл. каф. ЕОМ

Іванов Ю. С.

Львів 2022

Мета роботи: 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.

Варіант 20

y=tg(x)ctg(2x)

Лістинг програми:

FioApp.java

```
package KI34.Skalii.Lab6;
import java.util.Scanner;
import java.io.*;

/**
 * Class <code>EquationsApp</code> Implements driver for Equations class
 * @author Tetiana Skalii
 * @version 1.0
 */
public class FioApp{
    /**
     * @param args
     */
    public static void main(String[] args) throws FileNotFoundException, IOException
    {
        // TODO Auto-generated method stub
        CalcWFio obj = new CalcWFio();
        try (Scanner s = new Scanner(System.in)) {
                    System.out.print("Enter data: ");
                    double data = s.nextDouble();
                    obj.calculate(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());
    }
}
```

CalcWFio.java

```
package KI34.Skalii.Lab6;
import java.io.DataInputStream;
import java.io.DataOutputStream;
import java.io.File;
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.PrintWriter;
import java.util.Scanner;
/**
 * Class <code>CalcWFio</code> implements method for tg(x)*ctg(2x) expression
calculation and methods for writing and reading files
 * @author Tetiana Skalii
 * @version 1.0
 */

class CalcWFio
{

    /**
     * Method writes txt file
     * @param fName File's name
     * @throws FileNotFoundException error
     */

    public void writeResTxt(String fName) throws FileNotFoundException
    {
        PrintWriter f = new PrintWriter(fName);
        f.printf("%f ",result);
        f.close();
    }

    /**
     * Method read txt file
     * @param fName File's name
     */

    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());
        }
    }
    /**
     * Method writes bin file
     * @param fName File's name
     * @throws IOException error
     */
    public void writeResBin(String fName) throws FileNotFoundException, IOException
```

```
    {
        DataOutputStream f = new DataOutputStream(new FileOutputStream(fName));
        f.writeDouble(result);
        f.close();
    }
    /**
     * Method reads bin file
     * @param fName File's name
     * @throws IOException error
     */
    public void readResBin(String fName) throws FileNotFoundException, IOException
    {
        DataInputStream f = new DataInputStream(new FileInputStream(fName));
        result = f.readDouble();
        f.close();
    }

    public void calculate(double x)
    {
        Equations eq = new Equations();
        result = eq.calculate(x);
    }

    public double getResult()
    {
        return result;
    }
    private double result;
}
```

Equations.Java

```
package KI34.Skalii.Lab6;

/**
 * Class <code>Equations</code> implements method for y=tg(x)ctg(2x) expression
calculation
 * @author Tetiana Skalii
 * @version 1.0
 */
class Equations
{
      /**
     * Method calculates the tg(x)*ctg(2x) expression
     * @param x Angle in degrees
     * @throws CalcException
     */

    public double calculate(double x) throws CalcException
    {
        double y, rad;
        rad = x * Math.PI / 180.0;
        try
        {
            y = Math.tan(rad) * ( 1 / Math.tan(rad * 2));
            if (x==90 || x== -90 || x==0 || x==-180 || x==180 || 2*x == 90 || 2*x == -
90 || 2*x == 180 || 2*x == -180 )
                throw new ArithmeticException();
        }
        catch (ArithmeticException ex)
        {
            if (x==90 || x== -90 || x==-180 || x==180 || 2*x == 90 || 2*x == -90 || 2*x
== 180 || 2*x == -180 )
                throw new CalcException("Exception reason: Illegal value of X for
tangent calculation");
```

```
            else if (x==0)
                throw new CalcException("Exception reason: X = 0");
            else
                throw new CalcException("Unknown reason of the exception during
exception calculation");

        }
        return y;
    }

}
```

CalcException.java

```
package KI34.Skalii.Lab6;
/**
 * Class <code>CalcException</code> more precises ArithmeticException
 * @author Tetiana Skalii
 * @version 1.0
 */
class CalcException extends ArithmeticException
{

       private static final long serialVersionUID = 1L;

       public CalcException(){}

    public CalcException(String cause)
    {
        super(cause);
    }
}
```

Результат виконання програми:

```
<terminated> FioApp [Java Application] C:\Program Files\Java\jdk-18.0.2.1\bin\javaw.exe  (1 жовт. 2022 р., 20:29:10 – 20:29:15) [pid: 10592]
Enter data: 45
Exception in thread "main" KI34.Skalii.Lab6.CalcException: Exception reason: Illegal value of X for tangent calculation
        at KI34.Skalii.Lab6.Equations.calculate(Equations.java:23)
        at KI34.Skalii.Lab6.CalcWFio.calculate(CalcWFio.java:83)
        at KI34.Skalii.Lab6.FioApp.main(FioApp.java:21)
```

Виключення для неприпустимого значення для тангенса

```
Enter data: 0
Exception in thread "main" KI34.Skalii.Lab6.CalcException: Exception reason: X = 0
        at KI34.Skalii.Lab6.Equations.calculate(Equations.java:32)
        at KI34.Skalii.Lab6.CalcWFio.calculate(CalcWFio.java:83)
        at KI34.Skalii.Lab6.FioApp.main(FioApp.java:21)
```

Виключення при значенні нуля

```
<terminated> FioApp [Java Application] C:\Pro
Enter data: 60
Result is: -0.999999999999999
Result is: -0.999999999999999
Result is: -1.0
```

Рис.3.Результат виконання програми

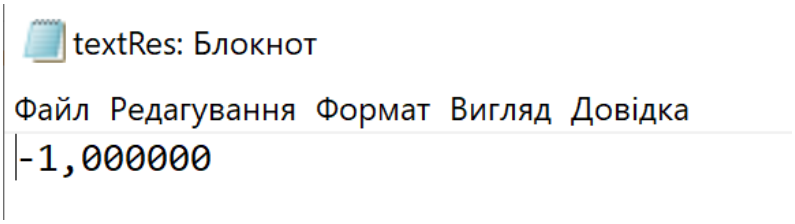

Рис.4.Успішний запис результату у текстовий файл

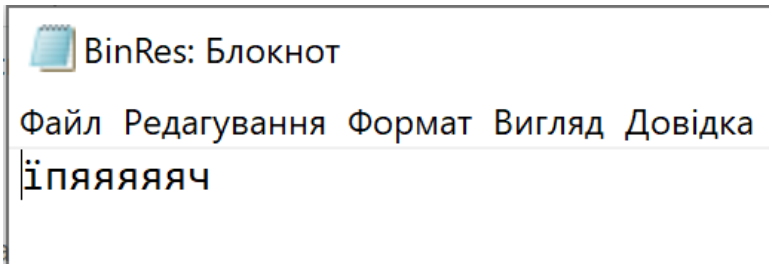

Рис.5.Успішний запис результату у бінарний файл

Згенерована документація

Package KI34.Skalii.Lab6

package KI34.Skalii.Lab6

All Classes and Interfaces | Classes | Exception Classes

Class	Description
CalcException	Class CalcException more precises ArithmeticException
CalcWFio	Class CalcWFio implements method for tg(x)*ctg(2x) expression calculation and methods for writing and reading files
Equations	Class Equations implements method for y=tg(x)ctg(2x)expression calculation
FioApp	Class FioApp Implements driver for CalcWFio class

Рис.6.Вмістиме вкладки Package

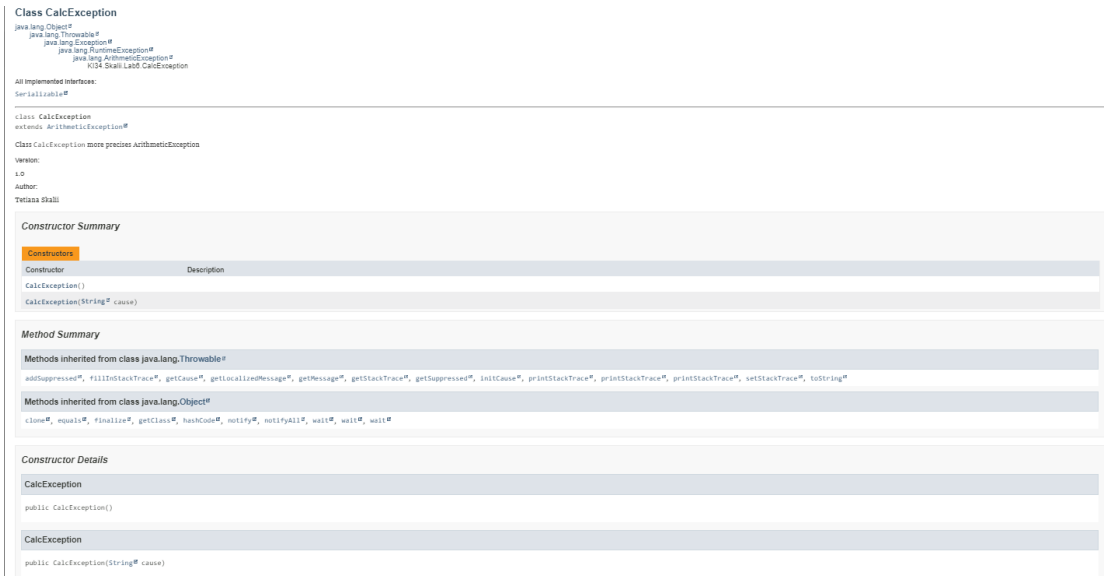

Рис.7.Вмістиме вкладки Class (CalcException)

Class CalcWFio

java.lang.Object
KI34.Skalii.Lab6.CalcWFio

```
class CalcWFio
extends Object
```

Class `CalcWFio` implements method for tg(x)*ctg(2x) expression calculation and methods for writing and reading files

Version:

1.0

Author:

Tetiana Skalii

Constructor Summary

Constructors

Constructor	Description
`CalcWFio()`	

Method Summary

All Methods | **Instance Methods** | **Concrete Methods**

Modifier and Type	Method	Description
void	`calculate(double x)`	
double	`getResult()`	
void	`readResBin(String fName)`	Method reads bin file
void	`readResTxt(String fName)`	Method read txt file
void	`writeResBin(String fName)`	Method writes bin file
void	`writeResTxt(String fName)`	Method writes txt file

Methods inherited from class java.lang.Object

`clone`, `equals`, `finalize`, `getClass`, `hashCode`, `notify`, `notifyAll`, `toString`, `wait`, `wait`, `wait`

Constructor Details

CalcWFio

```
CalcWFio()
```

Method Details

Рис.8.Вмістиме вкладки Class (CalcWFio)

Package KI34.Skalii.Lab6

Class Equations

java.lang.Object
    KI34.Skalii.Lab6.Equations

class **Equations**
extends Object

Class Equations implements method for y=tg(x)ctg(2x)expression calculation

Version:

1.0

Author:

Tetiana Skalii

Constructor Summary

Constructors

Constructor	Description
Equations()	

Method Summary

All Methods | **Instance Methods** | **Concrete Methods**

Modifier and Type	Method	Description
double	calculate(double x)	

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructor Details

Equations

```
Equations()
```

Method Details

calculate

```
public double calculate(double x)
                 throws CalcException
```

Throws:

CalcException

Рис.9.Вмістиме вкладки Class (Equations)

Class FioApp

java.lang.Object
    KI34.Skalii.Lab6.FioApp

```
public class FioApp
extends Object
```

Class FioApp Implements driver for CalcWFio class

Version:

1.0

Author:

Tetiana Skalii

Constructor Summary

Constructors

Constructor	Description
FioApp()	

Method Summary

All Methods | **Static Methods** | **Concrete Methods**

Modifier and Type	Method	Description
static void	main(String[] args)	

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructor Details

FioApp

```
public FioApp()
```

Method Details

main

```
public static void main(String[] args)
                throws FileNotFoundException,
                       IOException
```

Parameters:

args -

Throws:

FileNotFoundException

Рис.10.Вмістиме вкладки Class (FioApp)

Відповіді на контрольні запитання:

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

Для створення файлових потоків і роботи з ними у Java є 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. На відміну від, наприклад, мови програмування С, де для виконання усіх можливих операцій з файлами необхідно мати один вказівник на FILE у мові Java реалізовано інший набагато складніший і гнучкіший підхід, який дозволяє формувати такі властивості потоку, які найкраще відповідають потребам рішення конкретної задачі. Так у Java розділено окремі функціональні можливості потоків на різні класи. Компонуючи ці класи між собою і досягається необхідна кінцева функціональність потоку.

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.**
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, наприклад класом DataInputStream. 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, наприклад класом DataOutputStream. Інтерфейс DataOutput визначає такі методи для запису двійкових даних:

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

7. **Призначення класів DataInputStream і DataOutputStream.**

Класи DataOutputStream і DataInputStream дозволяють записувати і зчитувати дані примітивних типів.

Клас DataOutputStream представляє потік виведення і призначений для запису даних примітивних типів, таких, як int, double і т.д. Для запису кожного з примітивних типів призначений свій метод.

Клас DataInputStream діє протилежним чином - він зчитує з потоку дані примітивних типів. Відповідно для кожного примітивного типу визначено свій метод для зчитування.

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.**

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

Файли, що керуються класом RandomAccessFile, оснащені вказівником на позицію наступного байту, що має читатися або записуватися. Для того, щоб перемістити даний вказівник на довільну позицію в межах файлу використовується метод void seek(long pos). Параметр long pos визначає номер байту, що має читатися або записуватися.

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

Запис двійкових даних примітивних типів у потоки здійснюється за допомогою класів, що реалізують інтерфейс DataOutput, наприклад класом DataOutputStream.

Висновок:

На цій лабораторній роботі я оволоділа навиками використання засобів мови Java для роботи з потоками і файлами.