

Міністерство освіти і науки України
Національний університет „Львівська політехніка”



Звіт з лабораторної роботи №6
з дисципліни «Кросплатформні засоби програмування»
на тему: «ФАЙЛИ»

Виконав: ст.групи КІ-36

Борисов І. С.

Прийняв, та перевірів :

Іванов Ю. С.

Львів 2022

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.

Варіант №13

13. $y = \sin(x) / \text{ctg}(8x)$

Хід роботи:

Код програми:

Main.java

```
package LAB_06;

import java.io.FileNotFoundException;
import java.io.IOException;
import java.util.Scanner;

public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        try {

            String binFileName = "Lab_6.bin";
            String txtFileName = "Lab_6.txt";

            System.out.println("Enter X (in degrees): ");
            double y = Equation.calculate(scanner.nextDouble());

            CustomFileWriter.writeDoubleAsText(txtFileName, y);
            CustomFileWriter.writeDoubleAsBin(binFileName, y);

            double resTxt = CustomFileWriter.readDoubleAsText(txtFileName);
            double resBin = CustomFileWriter.readDoubleAsBin(binFileName);

            System.out.printf("True result: y = %f\n", y);
            System.out.printf("Txt file result: y_txt = %f\n", resTxt);
```

```

        System.out.printf("Bin file result: y_bin = %f\n", resBin);
        /**Виконується в будь якому випадку, незалежно від результатів
        виконання блоку try */

        /**ТИП 1 виключення */
    } catch (CalculationException calcException) {
        System.out.println(calcException.getMessage());
        /**ТИП 2 виключення */
    } catch (FileNotFoundException fileException) {
        System.out.printf("File exception was thrown with reason: %s\n",
fileException.getMessage());
        /**ТИП 3 виключення */
    } catch (IOException ioException) {
        System.out.printf("File exception was thrown with reason: %s\n",
ioException.getMessage());
    }
    finally {
        scanner.close();
    }
}
}

```

Equation.java

```

package LAB_06;

public class Equation {

    // метод повертає y=sin(x)/1/(8*tan(x)) як double

    public static double calculate(double xDegrees) throws CalculationException
    {
        //виключення вищого рівня з поясненням причини виникнення помилки
        if (xDegrees % 180.0d == 0.0d)
            throw new CalculationException("Degrees have illegal value for
equation: y=sin(x)/1/(8*tan(x))");

        double rad = Math.toRadians(xDegrees);
        double y = Math.sin(rad) / 1 / (8 * Math.tan(rad));
        //якщо результат не є числом, генеруємо виключення
        if (y == Double.NaN || Double.isInfinite(y))
            throw new CalculationException("Unknown reason caused throw during
calculations");

        return y;
    }
}

```

CustomFileWriter.java

```

package LAB_06;

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

public class CustomFileWriter {

    // Метод читає double як txt з файлу і повертає його

    public static double readDoubleAsText(String fileName) throws
FileNotFoundException {
        double res = 0.0d;

        Scanner inFile = new Scanner(new File(fileName));
        res = inFile.nextDouble();
    }
}

```

```

        inFile.close();

        return res;
    }

    // Метод записує double як текстові дані у файл

    public static void writeDoubleAsText(String fileName, double x) throws
FileNotFoundException {
        PrintWriter outFile = new PrintWriter(fileName);

        outFile.printf("%f", x);
        outFile.flush();
        outFile.close();
    }

    // метод зчитує double як bin з файлу і повертає його

    public static double readDoubleAsBin(String fileName) throws
FileNotFoundException, IOException {
        double res = 0.0d;

        DataInputStream inFile = new DataInputStream(new
FileInputStream(fileName));
        res = inFile.readDouble();
        inFile.close();

        return res;
    }

    // метод записує double як bin дані в файл

    public static void writeDoubleAsBin(String fileName, double x) throws
FileNotFoundException, IOException {
        DataOutputStream outFile = new DataOutputStream(new
FileOutputStream(fileName));

        outFile.writeDouble(x);
        outFile.flush();
        outFile.close();
    }
}

```

CalculationException.java

```

package LAB_06;

public class CalculationException extends ArithmeticException {

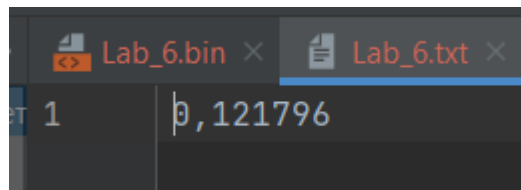
    public CalculationException() {
        super();
    }

    public CalculationException(String cause) {
        super(cause);
    }
}

```

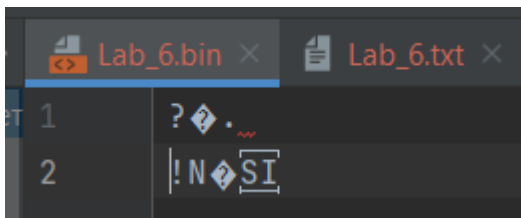
Результат виконання:

```
Enter X (in degrees): 13
True result: y = 0,121796
Txt file result: y_txt = 0,121796
Bin file result: y_bin = 0,121796
```



Lab_6.txt

1 0,121796



Lab_6.bin

1 ?
2 !N

Контрольні питання

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

Бібліотека класів мови Java має більше 60 класів для роботи з потоками. Потоками у мові Java називаються об'єкти з якими можна здійснювати обмін даними. Цими об'єктами найчастіше є файли, проте ними можуть бути стандартні пристрої вводу/виводу, блоки пам'яті і мережеві підключення тощо. Класи по роботі з потоками об'єднані у кілька ієрархій, що призначені для роботи з різними видами даних, або забезпечувати додаткову корисну функціональність, наприклад, підтримку ZIP архівів. Класи, що спадкуються від абстрактних класів `InputStream` і `OutputStream` призначені для здійснення байтового обміну інформацією. Підтримка мовою Java одиниць Unicode, де кожна одиниця має кілька байт, зумовлює необхідність у іншій ієрархії класів, що спадкується від абстрактних класів `Reader` і `Writer`. Ці класи дозволяють виконувати операції читання/запису не байтних даних, а двобайтних одиниць Unicode. Принцип здійснення читання/запису даних нічим не відрізняється від такого принципу у інших мовах програмування. Все починається з створення

потоків на запис або читання після чого викликаються методи, що здійснюють обмін інформацією. Після завершення обміну даними потоки необхідно закрити щоб звільнити ресурси.

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

5. Охарактеризуйте клас PrintWriter.

PrintWriter - це підклас Writer, який використовується для друку форматуваних даних у OutputStream або інший Writer, яким він керує. Усі методи PrintWriter не видають винятків I/O. Щоб перевірити, чи відбулося

виключення, можна викликати метод `checkError()`. При необхідності `PrintWriter` може виконувати автоматичне очищення (`flush`), це означає, що метод `flush()` буде викликаний відразу після виклику методу `println(..)` або під час друку тексту, що містить символ `'\n'`.

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`; • `writeChars`; • `writeBoolean`; • `writeUTF`.

7. Призначення класів `DataInputStream` і `DataOutputStream`.

`DataInputStream` - читання двійкових даних примітивних типів з потоків;

`DataOutputStream` - запис двійкових даних примітивних типів у потоки.

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).`

Параметр `mode` може приймати такі значення:

- `"r"` – читання;
- `"rw"` – читання/запис;
- `"rws"` – читання/запис даних з негайним синхронним записом змін у файл або метадані файлу;
- `"rwd"` – читання/запис даних з негайним синхронним записом змін у файл, метадані файлу не міняються одразу.

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

`DataOutputStream` реалізує інтерфейс `DataOutput`.

Висновок: на даній лабораторній роботі я оволодів навиками використання засобів мови Java для роботи з потоками і файлами.