Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Кафедра ЕОМ

Звіт

Лабораторна робота № 5

“ ФАЙЛИ У JAVA”

з курсу “Кросплатформні засоби програмування”

Варіант: 4

Виконав:

ст.гр.КІ-205

Воробець Тетяна

Прийняв:

доцент кафедри ЕОМ

Олексів М.В.

Львів 2024

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

Теоретичні відомості: Бібліотека класів мови Java має більше 60 класів для роботи з потоками. Потаками 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 потоку на запис або читання після чого викликаються методи, що здійснюють обмін інформацією. Після завершення обміну даними потоки необхідно закрити щоб звільнити ресурси.
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.
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 Systen.out.
Читання двійкових даних примітивних типів з потоків здійснюється за допомогою класів, що реалізують інтерфейс DataInput, наприклад класом DataInputStream.
Запис двійкових даних примітивних типів у потоки здійснюється за допомогою класів, що реалізують інтерфейс DataOutput, наприклад класом DataOutputStream.
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): RandomAccessFile(File file, String mode); RandomAccessFile(String name, String mode).

Завдання: 1. Створити клас, що реалізує методи читання/запису у текстовому і двійковому форматах результатів роботи класу, що розроблений у лабораторній роботі №4. (4. y=cos(x)/sin(x))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

Код програми:

Writer.java

```java
/**
 * Lab5
 */
package KI305.Vorobets.Lab5;

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;

/**
 * Writer is class for write and read result in .txt or .bin file
 */
public class Writer {

	private double result;

	/**
	 * Constructor
	 */

	public Writer() {

	}

	/**
	 * Constructor
	 * @param <code>result</code> result
	 */

	public Writer(double result) {
		this.result = result;
	}
```

```
	/**
	 * Method sets for result
	 */

	public void setResult(double result) {
		this.result = result;
	}

	/**
	 * Method gets for result
	 */

	public double getResult() {
		return this.result;
	}
	/**
	 * Method for write result in text file
	 */

	public void wrTxt(String fName) throws FileNotFoundException{
		PrintWriter f = new PrintWriter(fName);
		f.printf("%f ",result);
		f.close();
	}
	/**
	 * Method for read result in text file
	 */
	public double rdTxt(String fName){
		try{
			File f = new File (fName);
			if (f.exists()){
				Scanner s = new Scanner(f);
				result = s.nextDouble();
				s.close();
				return result;
			}
			else throw new FileNotFoundException("File " + fName +
"not found");
		}
		catch (FileNotFoundException ex){
			System.out.print(ex.getMessage());
		}
		return result;
	}
	/**
	 * Method for write result in bin file
	 */

	public void wrBin(String fName) throws FileNotFoundException,
IOException{
		DataOutputStream f = new DataOutputStream(new
FileOutputStream(fName));
		f.writeDouble(result);
```

```
            f.close();
      }
      /**
       * Method for read result in bin file
       */

      public double rdBin(String fName) throws FileNotFoundException,
IOException{
            DataInputStream f = new DataInputStream(new
FileInputStream(fName));
            result = f.readDouble();
            f.close();
            return result;
      }

}
```

Example.java

```
/**
 * Lab5
 */
package KI305.Vorobets.Lab5;


/**
 * Example is class implements example cos(x)/sin(x)
 */
public class Example {
      private int x;
      /**
       * Constructor
       */
      public Example() {
            this.x = 0;
      }
      /**
       * Constructor
       * @param <code>x</code> x is a radiant
       */
      public Example(int x) {
            this.x = x;
      }
      /**
       * Method is sets x in class
       * @param <code>x</code> x is a radiant
       */

      public void setX(int x) {
            this.x = x;
      }
      /**
       * Method calculation example
       */
```

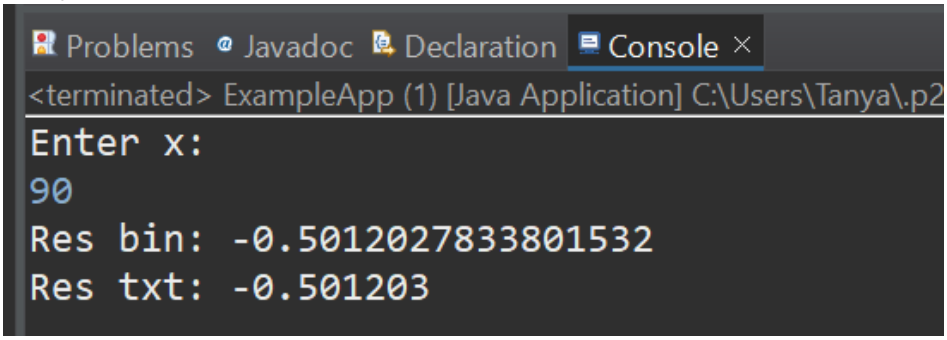
```
	public double calculate() throws ArithmeticException{
		if	(Math.sin(this.x) == 0) {
			throw new ArithmeticException("Error: divine to zero");
		}
		else {
			return Math.cos(this.x)/Math.sin(this.x);
		}
	}
}
```

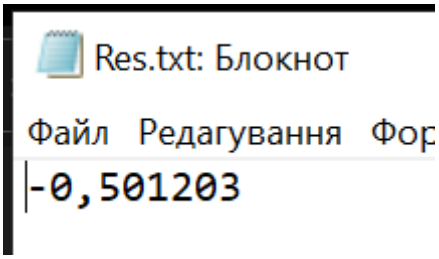
ExampleApp.java

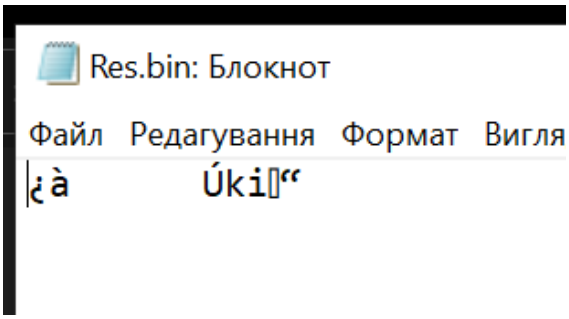
```
/**
 *
 */
package KI305.Vorobets.Lab5;

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

/**
 * ExampleApp is class-driver
 */
public class ExampleApp{

	/**
	 * @param args
	 */
	public static void main(String[] args) {
		// TODO Auto-generated method stub
			Example eq = new Example();
			System.out.println("Enter x: ");
			 try (Scanner in = new Scanner(System.in)) {
				eq.setX(in.nextInt());
				try{
					Writer wr = new Writer(eq.calculate());
					wr.wrBin("Res.bin");
					System.out.println("Res bin:
"+wr.rdBin("Res.bin"));
					wr.wrTxt("Res.txt");
					System.out.println("Res txt:
"+wr.rdTxt("Res.txt"));
				}
				catch (ArithmeticException ex){
					System.out.println(ex.getMessage());
				}
				catch (IOException ex) {
					System.out.println(ex.getMessage());
				}
			}
			catch (InputMismatchException ex) {
				System.out.println("Error: Wrong type number");
			}
	}
```

```
}
```

Результат:







Посилання на репозиторій:

https://github.com/NikaDe7/CPPT_Vorobets_TI_KI-35_1.git

Документація:

Constructors

Constructor	Description
`Writer()`	Constructor
`Writer(double result)`	Constructor

Method Summary

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

Modifier and Type	Method	Description
double	`getResult()`	Method gets for result
double	`rdBin(String fName)`	Method for read result in bin file
double	`rdTxt(String fName)`	Method for read result in text file
void	`setResult(double result)`	Method sets for result
void	`wrBin(String fName)`	Method for write result in bin file
void	`wrTxt(String fName)`	Method for write result in text file

Constructor Summary

Constructors

Constructor	Description
`Example()`	Constructor
`Example(int x)`	Constructor

Method Summary

All Methods | Instance Methods | Concrete Methods

Modifier and Type	Method	Description
double	`calculate()`	Method calculation example
void	`setX(int x)`	Method is sets x in class

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