Міністерство освіти і науки України

Національний університет “Львівська політехніка”

Кафедра ЕОМ



**Звіт**

З лабораторної роботи №5

Варіант№8

З дисципліни: «Кросплатформні засоби програмування»

На тему: «ФАЙЛИ У JAVA»

Виконав: ст. гр. КІ-304

Кісіль В. І.

Прийняв:

Олексів М.В.

Львів 2023

**Мета роботи:** 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.

**Принципи роботи з файловими потоками**

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. Так одні класи, як FileInputStream, забезпечують елементарний доступ до файлів, інші, як PrintWriter, надають додаткової функціональності по високорівневій обробці даних, що пишуться у файл. Ще інші, наприклад, BufferedInputStream забезпечують буферизацію. Таким чином, наприклад, щоб отримати буферизований файловий потік для читання інформації у форматі примітивних типів (char, int, double,…) слід створити потік з одночасним сумісним використанням функціональності класів FileInputStream, 4 BufferedInputStream і DataInputStream. Для цього слід здійснити наступний виклик:

DataInputStream din = new DataInputStream(

new BufferedInputStream(

new FileInputStream)));

Класи типу BufferedInputStream, DataInputStream, PushbackInputStream (дозволяє читати з потоку дані і повертати їх назад у потік) успадковані від класу FilterInputStream. Вони виступають так званими фільтрами, що своїм комбінуванням забезпечують додаткову лише необхідну функціональність при читанні даних з файлу. Аналогічний підхід застосовано і при реалізації класів для обробки текстових даних, що успадковані від Reader і Writer.

**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:**

**8. y=sin(x)/sin(2x-4)**

**Код програми:**

AppCalcWFio

package ki\_304.Kisil.Lab5;

import static java.lang.System.out;

import java.io.\*;

import java.util.\*;

public class AppCalcWFio {

/\*\*

\* @param args

\*/

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

{

// TODO Auto-generated method stub

Equations one = new Equations();

CalcWFio obj = new CalcWFio();

Scanner scan = new Scanner(System.in);

System.out.print("Enter data: ");

double data = scan.nextDouble();

try {

double result = one.calculate(data);

obj.insertInCalcWFio(result);

System.out.println("Result is: " + result);

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());

}

catch (CalcException ex)

{

// Блок перехоплює помилки обчислень виразу

out.print(ex.getMessage());

}

}

}

CalcWFio

package ki\_304.Kisil.Lab5;

import java.io.\*;

import java.util.\*;

public class CalcWFio {

public void writeResTxt(String fName) throws FileNotFoundException

{

PrintWriter f = new PrintWriter(fName);

f.printf("%f ",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 insertInCalcWFio(double x)

{

result = x;

}

public double getResult()

{

return result;

}

private double result;

}

Equations

package ki\_304.Kisil.Lab5;

public class Equations {

/\*\*

\* Method calculates the ((2 / tg(x)) / x) expression

\* @param <code>x</code> Angle in degrees

\* @throws CalcException

\*/

public double calculate(double x) throws CalcException

{

double y, rad, rad1;

rad = x \* Math.PI / 180.0;

rad1=(2\*x-4)\* Math.PI / 180.0;

try

{

y = Math.sin(rad) / Math.sin(rad1);

// 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)

{

// створимо виключення вищого рівня з поясненням причини

// виникнення помилки

if (Math.sin(rad1)==0)

throw new CalcException("Exception reason: denominator = 0");

else

throw new CalcException("Unknown reason of the exception during exception calculation");

}

return y;

}

}

class CalcException extends ArithmeticException

{

public CalcException(){}

public CalcException(String cause)

{

super(cause);

}

}

**Результат роботи програми:**

Enter data: 5

Result is: 0.8337991397088556

Result is: 0.8337991397088556

Result is: 0.833799

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