Національний університет «Львівська Політехніка»

Інститут комп’ютерних технологій, автоматики та метрології

Кафедра електронних обчислювальних машин



Звіт

Про виконання лабораторної роботи №5

З дисципліни «Кросплатформлені засоби програмування»

**Виконав:**

студент групи КІ-305

Ключко Д.С.

**Перевірив:**

Доцент кафедри ЕОМ

Іванов Ю. С.

Львів – 2023

**Тема:** Файли у java.

**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

**Завдання згідно варіанту:**

**Варіант: 12**

****

**Виконання:**

**Код програми:**

package lab5Klyuchko305;

import java.util.Scanner;

import java.io.\*;

import static java.lang.System.out;

/\*\*

\* Class <code>EquationsApp</code> Implements driver for Equations class

\*

\* @author Klyuchko Dmytro

\* @version 1.0

\*/

public class MathApp {

/\*\*

\* @param args

\* @throws IOException

\*/

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

Scanner in = new Scanner(System.in);

double result;

try {

int inputX;

out.print("Введіть значення x: ");

inputX = in.nextInt();

calcEquations eq = new calcEquations();

result = eq.calculate(inputX);

ReadWritetxt Workfile = new ReadWritetxt();

ReadWritebin Workfile1 = new ReadWritebin();

Workfile.WriteTxt(result);

Workfile1.WriteBin(result);

Workfile1.ReadBin();

try {

Workfile.ReadTxt();

// Блок перехоплює помилки роботи з файлом

} catch ( FileNotFoundException ex) {

System.out.println("Файл не знайдено");

}

// Блок перехоплює помилки обчислень виразу

} catch (CalcException ex) {

out.print(ex.getMessage());

}

}

}

/\*\*

\* Class <code>ReadWritetxt</code> work with text files

\* expression

\* calculation

\*

\* @author Klyuchko Dmytro

\* @version 1.0

\*/

class ReadWritetxt{

private PrintWriter fout;

public ReadWritetxt()

{

try {

fout = new PrintWriter("MyFile.txt");

} catch (FileNotFoundException e) {

out.print("Файл за даним шляхом відсутній !");

}

}

/\*\*

\* Method write result to text file

\*

\* @param <code>fName</code> Name of the file

\* @throws

\*/

public void WriteTxt(double result)

{

fout.print("Результат обчислення функції y = sin(x) / tg(4x) = ");

fout.print(result);

fout.flush();

fout.close();

}

/\*\*

\* Method read result from text file

\*

\* @param <code>fName</code> Name of the file

\* @throws

\*/

public void ReadTxt() throws FileNotFoundException

{

BufferedReader br = null;

br = new BufferedReader(new FileReader("MyFile.txt"));

String line;

try {

while ((line = br.readLine()) != null)

{

out.println(line);

}

br.close();

} catch (IOException e) {

e.printStackTrace();

}

}

}

/\*\*

\* Class <code>ReadWritebin</code> work with bin files

\* expression

\* calculation

\*

\* @author Klyuchko Dmytro

\* @version 1.0

\*/

class ReadWritebin{

/\*\*

\* Method write result to binary file

\*

\* @param <code>fName</code> Name of the file

\* @throws

\*/

public void WriteBin(double result) throws FileNotFoundException, IOException

{

DataOutputStream f = new DataOutputStream(new FileOutputStream("MyFile.bin"));

f.writeDouble(result);

f.close();

}

/\*\*

\* Method read result from binary file

\*

\* @param <code>fName</code> Name of the file

\* @throws

\*/

public void ReadBin() throws FileNotFoundException, IOException

{

double line;

DataInputStream f = new DataInputStream(new FileInputStream("MyFile.bin"));

line = f.readDouble();

out.println(line);

f.close();

}

}

/\*\*

\* Class <code>CalcException</code> more precises ArithmeticException

\*

\* @author Klyuchko Dmytro

\* @version 1.0

\*/

class CalcException extends Exception {

public CalcException() {

}

public CalcException(String cause) {

super(cause);

}

}

/\*\*

\* Class <code>calcEquations</code> implements method for y=sin(x)/tg(4x)

\* expression

\* calculation

\*

\* @author Klyuchko Dmytro

\* @version 1.0

\*/

class calcEquations {

/\*\*

\* Method calculates the y=sin(x)/tg(4x) expression

\*

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

\* @throws CalcException

\*/

public double calculate(int x) throws CalcException {

double y, rad, rad1;

rad = x \* Math.PI / 180.0;

rad1 = 4\*rad;

// створимо виключення вищого рівня з поясненням причини

// виникнення помилки

try {

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

if (Double.isNaN(y) || Double.isInfinite(y) || x == 90 || x == -90) {

throw new CalcException("Причина помилки: некоретне значення кута для обчислення тангенсу!");

}

} catch (ArithmeticException ex) {

if (x == 0) {

throw new CalcException("Причина помилки: x = 0");

} else {

throw new CalcException("Незрозуміла помилка при обрахунках!");

}

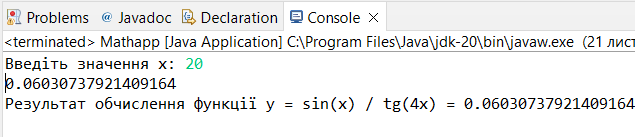
}

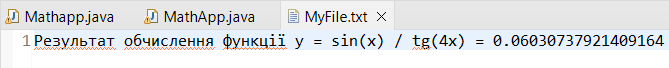
return y;

}

}

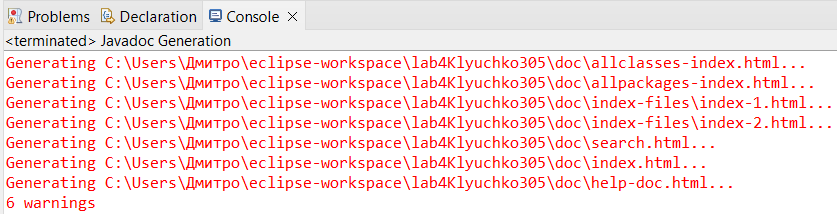
**Результат роботи програми у консолі та файлі:**

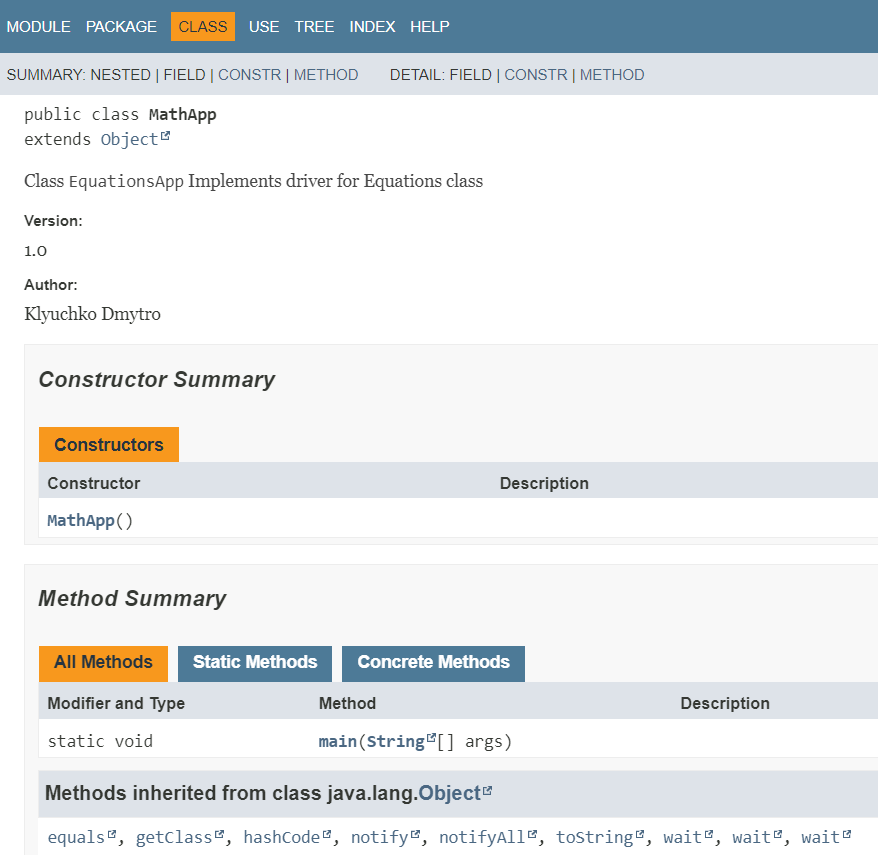
****

****

****

**Фрагмент генерації документації:**

****

****

**Контрольні запитання**

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

**Відповідь:**

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.

1. Охарактеризуйте клас Scanner.

**Відповідь:**

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.

1. Наведіть приклад використання класу Scanner

**Відповідь:**

Приклад читання даних за допомогою класу Scanner з стандартного потоку вводу:

****

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

**Відповідь:**

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.

1. Охарактеризуйте клас PrintWriter

**Відповідь:**

Клас PrintWriter в Java використовується для запису послідовності символів у текстовий файл. Він дозволяє використовувати методи для друку даних в текстовий потік та записувати їх у файл.

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

**Відповідь:**

У мові програмування Java для роботи з двійковими даними існує низка класів, які дозволяють зчитувати та записувати дані у двійковому форматі. Найбільш поширеними з них є класи DataInputStream і DataOutputStream.

1. Призначення класів DataInputStream і DataOutputStream.**Відповідь:**

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.

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

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

1. Охарактеризуйте клас RandomAccessFile

**Відповідь:**

Клас **RandomAccessFile** в Java дозволяє читати та записувати дані у файлі в режимі випадкового доступу. Цей клас надає можливість переміщати покажчик файлу на конкретну позицію та зчитувати чи записувати дані з цієї позиції.

Основні особливості **RandomAccessFile**:

1. **Доступ до файлу за позицією**: Він дозволяє читати або записувати дані у вказану позицію в файлі, що надає більш гнучкий та точний доступ до даних.
2. **Підтримка читання та записування примітивних типів даних**: Клас має методи для зчитування та запису цілих чисел, рядків, дійсних чисел, символів тощо.
3. **Режими роботи з файлом**: Підтримується як режим читання, так і режим запису, а також можливість працювати в режимі, який дозволяє одночасно читати та записувати дані в файл.
4. **Потокозалежність**: Клас працює зі зміщеннями у байтах у файлі, що робить його потенційно швидким, але при цьому може призвести до складнощів у роботі з рядками або іншими типами даних, що не мають фіксованого розміру.
5. Який зв’язок між інтерфейсом DataOutput і класом DataOutputStream?

**Відповідь:**

Інтерфейс DataOutput та клас DataOutputStream пов'язані один з одним через взаємозв'язок реалізації.

DataOutput - це інтерфейс у пакеті java.io, який визначає методи для запису примітивних типів даних у потік байтів.

DataOutputStream - це клас у пакеті java.io, який реалізує інтерфейс DataOutput. Він надає можливість записувати примітивні типи даних у потік байтів.

Отже, клас DataOutputStream реалізує функціонал інтерфейсу DataOutput, що означає, що він надає реалізацію всіх методів, оголошених в інтерфейсі DataOutput. Коли ви використовуєте DataOutputStream, ви можете використовувати методи, які оголошені в DataOutput, для запису даних у вихідний потік. Це дозволяє вам зручно і безпечно записувати примітивні дані у вихідний потік байтів, наприклад, у файл чи мережевий потік.

**Висновок:** на цій лабораторній роботі я оволодів навиками використання засобів мови Java для роботи з потоками і файлами. Також модернізував програму з попередньої лабораторної роботи таким чином, щоб вона записувала і зчитувала результат з бінарного і текстового файлів.