Міністерство освіти і науки України

Національний університет «Львівська політехніка»



Звіт

до лабораторної роботи №4

з дисципліни: “Кросплатформні засоби програмування”

на тему: «Виключення»

Варіант №12

Виконала:

ст.гр. КІ-307

Коваль К. О.

Прийняв:

Іванов Ю. С.

**Львів 2023**

**Мета:** оволодіти навиками використання механізму виключень при написанні програм мовою Java.

**Завдання:**

1. Створити клас, що реалізує метод обчислення виразу заданого варіантом. Написати на мові Java та налагодити програму-драйвер для розробленого класу. Результат обчислень записати у файл. При написанні програми застосувати механізм виключень для виправлення помилкових ситуацій, що можуть виникнути в процесі виконання програми. Програма має розміщуватися в пакеті Група.Прізвище.Lab4 та володіти коментарями, які дозволять автоматично згенерувати документацію до розробленого пакету.

2. Автоматично згенерувати документацію до розробленого пакету.

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:

1. Дайте визначення терміну «виключення».

2. У яких ситуаціях використання виключень є виправданим?

3. Яка ієрархія виключень використовується у мові Java?

4. Як створити власний клас виключень?

5. Який синтаксис оголошення методів, що можуть генерувати виключення?

6. Які виключення слід вказувати у заголовках методів і коли?

7. Як згенерувати контрольоване виключення?

8. Розкрийте призначення та особливості роботи блоку try.

9. Розкрийте призначення та особливості роботи блоку catch.

10. Розкрийте призначення та особливості роботи блоку finally.

**Варіант 12: y=sin(x)/tg(4x)**

**Виконання:**

**Lab4KovalKI307.java:**

package KI307.Koval.Lab4;

import java.util.Scanner;

import java.io.\*;

import static java.lang.System.out;

/\*\*

\* Class <code>Lab4KovalKI307</code> Implements driver for Equations class

\* @author Koval Kateryna

\* @version 1.2

\*/

public class Lab4KovalKI307 {

/\*\*

\* @param args

\* @throws Exception

\*/

public static void main(String[] args) throws Exception {

try

{

out.print("Enter file name: ");

Scanner in = new Scanner(System.in);

String fName = in.nextLine();

PrintWriter fout = new PrintWriter(new File(fName));

try

{

try

{

Equations eq = new Equations();

out.print("Enter X: ");

fout.print(eq.calculate(in.nextInt()));

}

finally

{

// Цей блок виконається за будь-яких обставин

out.println("Result is written to file '"+ fName + "'.");

fout.flush();

fout.close();

}

}

catch (CalcException ex)

{

// Блок перехоплює помилки обчислень виразу

out.print(ex.getMessage());

}

}

catch (FileNotFoundException ex)

{

// Блок перехоплює помилки роботи з файлом навіть якщо вони

// виникли у блоці finally

out.print("Exception reason: Perhaps wrong file path");

}

}

}

/\*\*

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

\* @author Koval Kateryna

\* @version 1.2

\*/

class CalcException extends ArithmeticException

{

public CalcException(){}

public CalcException(String cause)

{

super(cause);

}

}

/\*\*

\* Class <code>Equations</code> implements method for (sin(x)/tg(4x)) expression calculation

\* @author Koval Kateryna

\* @version 1.2

\*/

class Equations

{

/\*\*

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

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

\* @throws CalcException

\*/

public double calculate(int x) throws CalcException

{

double y, rad;

rad = x \* Math.PI / 180.0;

try

{

y = Math.sin(rad)/Math.tan(4\*rad);

// 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 (rad==Math.PI/2.0 || rad==-Math.PI/2.0)

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");

}

out.println("y = "+ y);

return y;

}

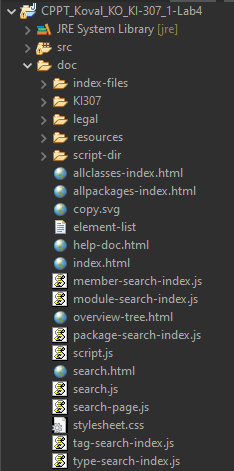
}

**Результат:**

****

****

**Документація:**

****

**Відповіді на контрольні запитання:**

1. \*\*Визначення терміну «виключення»:\*\*

- Виключення (або помилка) в програмуванні вказує на несподівану або некоректну ситуацію, яка виникає під час виконання програми.

2. \*\*Ситуації використання виключень:\*\*

- Обробка помилок, які можуть виникнути під час виконання програми.

- Відділення коду, який може викликати помилку, від коду обробки цієї помилки.

3. \*\*Ієрархія виключень в Java:\*\*

- Всі виключення у Java є підкласами класу `Throwable`.

- Два основних підкласи: `Exception` (виникає в результаті помилок програмування) та `Error` (виникає в результаті серйозних проблем, як правило, не пов'язаних з програмним кодом).

4. \*\*Створення власного класу виключень:\*\*

- Створення класу, який наслідується від `Exception` або `RuntimeException`.

```java

public class CustomException extends Exception {

// конструктори та інші методи

}

```

5. \*\*Синтаксис оголошення методів, що можуть генерувати виключення:\*\*

```java

access\_modifier return\_type methodName() throws ExceptionType {

// тіло методу

}

```

6. \*\*Вказання виключень у заголовках методів:\*\*

- Вказується ключове слово `throws`, за яким слідує перелік виключень, які метод може генерувати.

7. \*\*Генерація контрольованого виключення:\*\*

- Виклик конструктора класу виключення та його відправлення за допомогою ключового слова `throw`.

8. \*\*Блок `try` в Java:\*\*

- Використовується для обгортання коду, який може викликати виключення.

- Всередині блоку `try` розміщується код, який слід моніторити на виникнення виключень.

9. \*\*Блок `catch` в Java:\*\*

- Використовується для обробки виняткових ситуацій, які можуть виникнути в блоку `try`.

- Може бути кілька блоків `catch` для різних типів виключень.

10. \*\*Блок `finally` в Java:\*\*

- Використовується для виконання коду, який завжди повинен виконуватися, незалежно від того, чи виникали виключення, чи ні.

**Висновок:** на даній лабораторній роботі я оволоділа навиками використання механізму виключень при написанні програм мовою Java.