Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Кафедра «Електронних обчислювальних машин»



**Звіт**

з лабораторної роботи № 4

з дисципліни: «Кросплатформенні засоби програмування»

**На тему:** «Виключення»

**Виконав:**

студент групи КІ-306

Бокало П.М.

**Прийняв:**

доцент кафедри ЕОМ

Іванов Ю. С.

Львів – 2023

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

**Завдання(варіант №2):**

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.

**Індивідуальне завдання:**



**Вихідний код програми:**

**Файл KI306.Bokalo.Lab4.java:**

package KI306.Bokalo.Lab4;

import java.io.FileNotFoundException;

import java.io.PrintWriter;

import java.util.Scanner;

/\*\*

\* Дана програма є класом драйвером який тестує роботу класу Calco

\* @author Petro Bokalo KI-306

\* @version 1.0

\* @since version 1.0

\*/

public class Lab4BokaloKI306

{

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

{

double x = 0;

Scanner input = new Scanner(System.in);

System.out.print("Input your X : ");

if(input.hasNextDouble())

{

x = input.nextDouble();

}else

{

System.out.print("Invalid data inputed \n");

}

Calco c = new Calco(x);

try

{

System.out.print("Ctg of x = " + c.result() + "\n");

}

catch(ArithmeticException ex)

{

System.out.print(ex.getMessage());

}

c.in\_file();

c.close\_file();

input.close();

}

}

**Файл Calco.java**

package KI306.Bokalo.Lab4;

import java.io.BufferedWriter;

import java.io.FileWriter;

import java.io.IOException;

import java.io.PrintWriter;

/\*\*

\* Тут реалізований клас Calco який рахує Ctg введеного числа

\* @author Petro Bokalo KI-306

\* @version 1.0

\* @since version 1.0

\*/

public class Calco

{

private double x;

private double ctg = 0;

PrintWriter fout;

public Calco(double x)

{

this.x = x;

}

public double result() throws ArithmeticException

{

if(Math.tan(x) == 0)

{

throw new ArithmeticException("Exeption : Sin is equal to 0!!!\n");

}

else

{

ctg = 1 / Math.tan(x);

}

return ctg;

}

public void in\_file()

{

try

{

fout = new PrintWriter(new BufferedWriter(new FileWriter("Lab4.txt")));

fout.print("Ctg of x = " + result() + "\n");

} catch (IOException e)

{

System.err.println("Can't use the file!!!\n");

}

catch (ArithmeticException a)

{

System.out.print(a.getMessage());

}

}

public void close\_file()

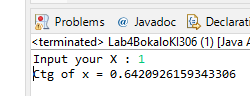
{

fout.close();

}

}

**Результат роботи програми:**



**Вміст файлів :**

**Файл Lab4.txt:**

Ctg of x = 0.6420926159343306

**Фрагмент згенерованої документації:**



**Відповідь на контрольні питання**

1. Дайте визначення терміну «виключення».

- Виключення (або exception) - це об'єкт, який виникає під час виконання програми і вказує на помилку або надзвичайну ситуацію.

2. У яких ситуаціях використання виключень є виправданим?

- Виключення використовуються для обробки помилок та надзвичайних ситуацій, коли неможливо нормально виконати програму. Їх використання допомагає відстежувати, відловлювати і обробляти помилки без припинення виконання програми.

3. Яка ієрархія виключень використовується у мові Java?

- У мові Java існує ієрархія класів виключень, де базовий клас - `java.lang.Throwable`, а дві основні гілки це `java.lang.Error` і `java.lang.Exception`. Остання гілка поділяється на контрольовані (checked) і неконтрольовані (unchecked) виключення.

4. Як створити власний клас виключень?

- Для створення власного класу виключень потрібно створити клас, який наслідується від `java.lang.Exception` або його підкласу.

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

- Синтаксис оголошення методу, що може генерувати виключення: `public void methodName() throws SomeException`.

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

- Слід вказувати контрольовані (checked) виключення у заголовках методів, якщо метод може генерувати це виключення або його підкласи. Неконтрольовані (unchecked) виключення не обов'язково вказувати.

7. Як згенерувати контрольоване виключення?

- Контрольоване виключення генерується за допомогою ключового слова `throw` в коді методу, наприклад: `throw new SomeException("Повідомлення про помилку")`.

8. Розкрийте призначення та особливості роботи блоку try.

- Блок `try` використовується для оточення коду, який може генерувати виключення. Він спробує виконати цей код, і якщо виникає виключення, керування передається блокам `catch` або `finally`.

9. Розкрийте призначення та особливості роботи блоку catch.

- Блок `catch` використовується для обробки виключень, які були згенеровані в блоку `try`. Він приймає параметр, який вказує на тип оброблюваного виключення.

10. Розкрийте призначення та особливості роботи блоку finally.

- Блок `finally` використовується для коду, який завжди виконується, незалежно від того, чи були виключення, чи ні. Він використовується для виконання завершальних операцій, таких як закриття файлів чи звільнення ресурсів.

**Висновок:** на цій лабораторній роботі, я ознайомився з виключеннями в мові програмування java. Написав програму згідно до свого варіанту. Навчився робити контрольовані виключення та ловити їх.