Міністерство освіти і науки України

Національний університет “Львівська політехніка”

Кафедра ЕОМ

Звіт

до лабораторної роботи №4

З дисципліни: «Кросплатформні засоби програмування»

На тему: «ВИКЛЮЧЕННЯ»

Виконала: ст. гр. КІ-305

Романюк М. Р.

Прийняв: доцент кафедри ЕОМ

Іванов Ю. С.

Львів 2023

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

ТЕОРЕТИЧНІ ВІДОМОСТІ

Виключення – це механізм мови Java, що забезпечує негайну передачу керування блоку коду опрацювання критичних помилок при їх виникненні уникаючи процесу розкручування стеку. Генерація виключень застосовується при:

- *помилках введення*, наприклад, при введенні назви неіснуючого файлу або Інтернет адреси з подальшим зверненням до цих ресурсів, що призводить до генерації помилки системним програмним забезпеченням;
- *збоях обладнання;*
- *помилках, що пов'язані з фізичними обмеженнями комп'ютерної системи*, наприклад, при заповненні оперативної пам'яті або жорсткого диску;
- *помилках програмування*, наприклад, при некоректній роботі методу, читанні елементів порожнього стеку, виходу за межі масиву тощо.

Мій варіант №22: y=tg(4x)/x

Код програми:

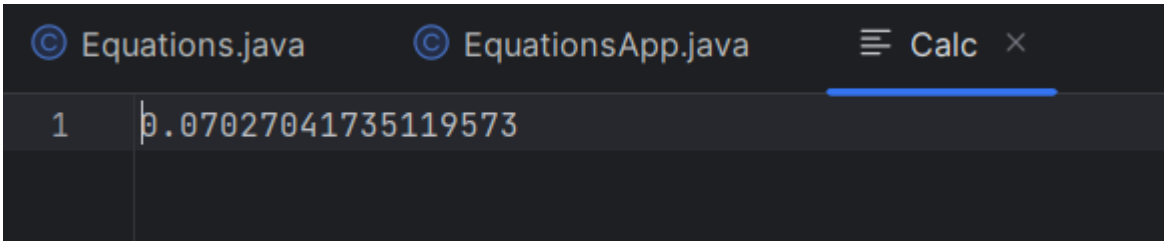
Файл Equations.java

```
package Ki305.Romaniuk.Lab4;

/**
 * Клас Equations реалізує метод для обчислення виразу y = tg(4x) / x.
 */
class Equations {

    /**
     * Метод обчислює вираз y = tg(4x) / x.
     *
     * @param x Кут в градусах.
     * @return Результат обчислення виразу.
     * @throws CalcException Виняток, який виникає при некоректних обчисленнях.
     */
    public double calculate(int x) throws CalcException {
        double y, rad;
        rad = 4 * x * Math.PI / 180.0; // Обчислення кута в радіанах
        try {
            y = Math.tan(rad) / x; // Обчислення tg(4x) / x
            if (Double.isNaN(y) || Double.isInfinite(y) || x == 0) {
                throw new ArithmeticException();
            }
        } catch (ArithmeticException ex) {
            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");
            }
        }
        return y;
    }
}
```

Файл EquationsApp.java

```java
package Ki305.Romaniuk.Lab4;

import java.util.Scanner;
import java.io.*;
import static java.lang.System.out;

/**
 * Клас EquationsApp є програмою-драйвером для обчислення виразу y = tg(4x) / x та запису результату в файл.
 * Користувач вводить ім'я файлу та значення X, програма обчислює результат та записує його в файл.
 */
public class EquationsApp {

    /**
     * Головний метод програми, який виконує основну логіку.
     * @param args Аргументи командного рядка (не використовуються).
     */
    public static void main(String[] args) {
        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: ");
                    int x = in.nextInt();
                    double result = eq.calculate(x);
                    fout.print(result);
                } finally {
                    fout.flush();
                    fout.close();
                }
            } catch (CalcException ex) {
                out.print(ex.getMessage());
            }
        } catch (FileNotFoundException ex) {
            out.print("Exception reason: Perhaps wrong file path");
        }
    }
}

/**
 * Клас CalcException є підкласом ArithmeticException і використовується для обробки помилок обчислення виразу.
 */
class CalcException extends ArithmeticException {

    /**
     * Конструктор без параметрів.
     */
    public CalcException() {
    }

    /**
     * Конструктор з параметром, який приймає причину помилки.
     * @param cause Причина помилки.
     */
    public CalcException(String cause) {
        super(cause);
    }
}
```

Результати роботи програми:



Згенерована документація:

All Methods | Static Methods | Concrete Methods

Modifier and Type	Method	Description
static void	main(String[] args)	Головний метод програми, який виконує основну логіку.

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructor Details

EquationsApp

```
public EquationsApp()
```

Method Details

main

```
public static void main(String[] args)
```

Головний метод програми, який виконує основну логіку.

Parameters:

`args` - Аргументи командного рядка (не використовуються).

PACKAGE | CLASS | TREE | INDEX | HELP

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD SE

Package Ki305.Romaniuk.Lab4

Class EquationsApp

java.lang.Object
  Ki305.Romaniuk.Lab4.EquationsApp

```
public class EquationsApp
extends Object
```

Клас EquationsApp є програмою-драйвером для обчислення виразу y = tg(4x) / x та запису результату в файл. Користувач вводить ім'я файлу та значення X, пр
записує його в файл.

Constructor Summary

Constructors

Constructor	Description
EquationsApp()	

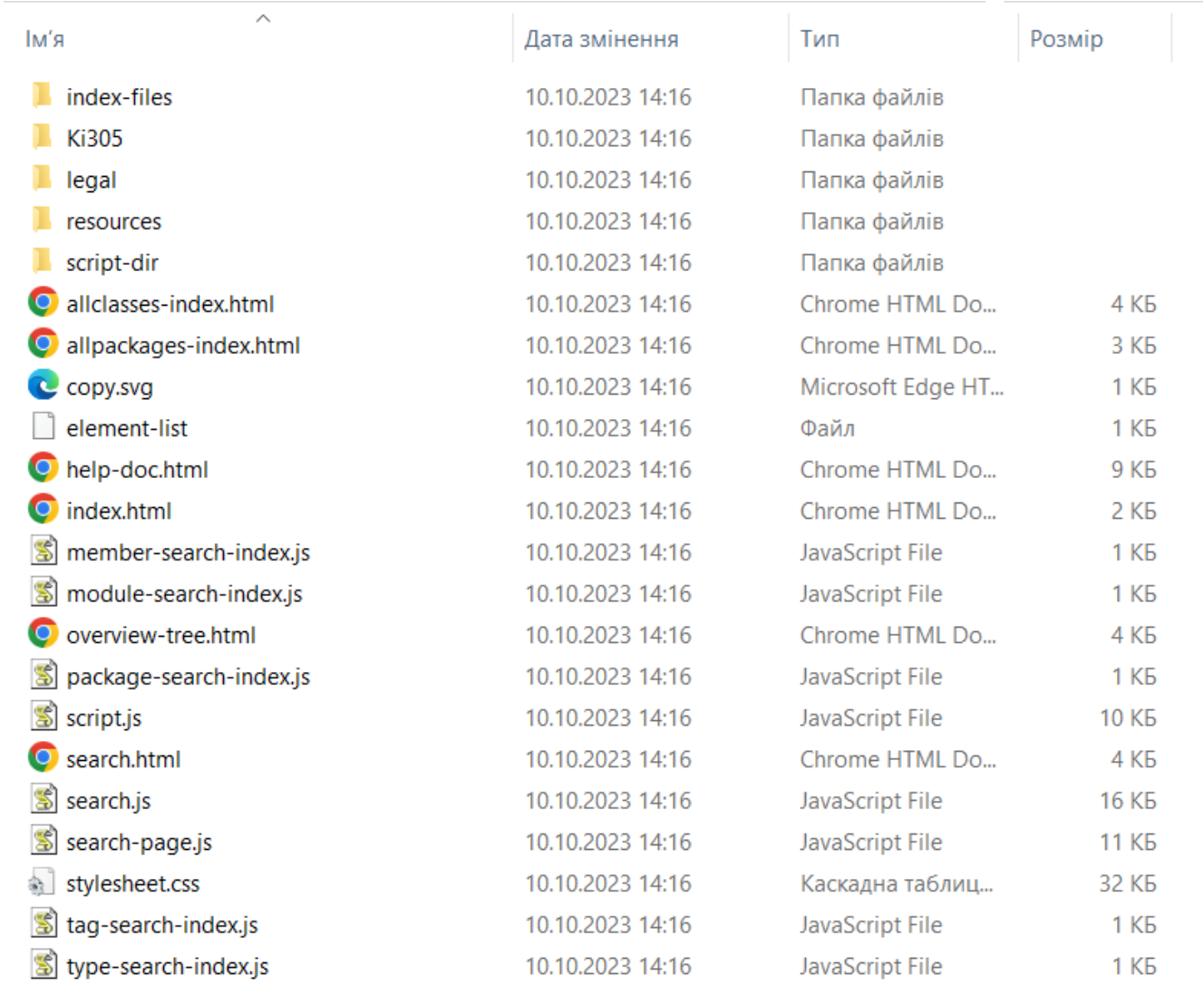
Method Summary

All Methods | Static Methods | Concrete Methods

Modifier and Type	Method	Description
static void	main(String[] args)	Головний метод програми, який виконує основну логіку.

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait



Ім'я	Дата змінення	Тип	Розмір
index-files	10.10.2023 14:16	Папка файлів	
Ki305	10.10.2023 14:16	Папка файлів	
legal	10.10.2023 14:16	Папка файлів	
resources	10.10.2023 14:16	Папка файлів	
script-dir	10.10.2023 14:16	Папка файлів	
allclasses-index.html	10.10.2023 14:16	Chrome HTML Do...	4 КБ
allpackages-index.html	10.10.2023 14:16	Chrome HTML Do...	3 КБ
copy.svg	10.10.2023 14:16	Microsoft Edge HT...	1 КБ
element-list	10.10.2023 14:16	Файл	1 КБ
help-doc.html	10.10.2023 14:16	Chrome HTML Do...	9 КБ
index.html	10.10.2023 14:16	Chrome HTML Do...	2 КБ
member-search-index.js	10.10.2023 14:16	JavaScript File	1 КБ
module-search-index.js	10.10.2023 14:16	JavaScript File	1 КБ
overview-tree.html	10.10.2023 14:16	Chrome HTML Do...	4 КБ
package-search-index.js	10.10.2023 14:16	JavaScript File	1 КБ
script.js	10.10.2023 14:16	JavaScript File	10 КБ
search.html	10.10.2023 14:16	Chrome HTML Do...	4 КБ
search.js	10.10.2023 14:16	JavaScript File	16 КБ
search-page.js	10.10.2023 14:16	JavaScript File	11 КБ
stylesheet.css	10.10.2023 14:16	Каскадна таблиц...	32 КБ
tag-search-index.js	10.10.2023 14:16	JavaScript File	1 КБ
type-search-index.js	10.10.2023 14:16	JavaScript File	1 КБ

Відповіді на контрольні запитання

1. Дайте визначення терміну «виключення».

Визначення виключення: Виключення (або ексепшн) - це об'єкт, який виникає в програмі в результаті помилки або некоректної ситуації і призводить до припинення нормального виконання програми.

2. У яких ситуаціях використання виключень є виправданим?

Ситуації для використання виключень: Виключення виправдано використовувати в таких ситуаціях:

- При обробці помилок та некоректних ситуацій.
- Коли виникає необхідність повідомити вищестоячому коду про помилку.
- Для забезпечення гнучкої обробки помилок і відновлення програми.

3. Яка ієрархія виключень використовується у мові Java?

Ієрархія виключень у мові Java: У Java існує ієрархія виключень, де клас **Throwable** є батьківським класом для усіх виключень. Дві основні гілки цієї ієрархії - це **Error** (помилки, з якими програма зазвичай не повинна взаємодіяти) і **Exception** (виключення, які програма може обробляти).

4. Як створити власний клас виключень?

Створення власного класу виключень: Для створення власного класу виключень потрібно створити клас, який успадковується від класу **Exception** (або іншого класу виключень, якщо це підходить) та може містити конструктори та методи для

налаштування та отримання додаткової інформації про помилку.
5. Який синтаксис оголошення методів, що можуть генерувати виключення?
Синтаксис оголошення методів, що можуть генерувати виключення: Можна оголосити метод, який може генерувати виключення, за допомогою ключового слова **throws** в його заголовку. Наприклад:
public void myMethod() throws MyException { // Код методу }
6. Які виключення слід вказувати у заголовках методів і коли?
Виключення у заголовках методів і коли: Виключення слід вказувати у заголовках методів, які можуть породжувати помилки або які використовують методи, що генерують виключення. Це допомагає іншим програмістам розуміти, які помилки можуть виникнути в методі і як їх обробляти.
7. Як згенерувати контрольоване виключення?
Генерація контрольованого виключення: Для генерації контрольованого виключення ви повинні використовувати ключове слово throw разом із створенням об'єкту виключення та його викиданням. Наприклад:
throw new MyException("Це моя власна помилка");
8. Розкрийте призначення та особливості роботи блоку try.
Блок try: Блок **try** використовується для обрамлення коду, який може викидати виключення. Якщо виникає виключення, виконання коду в блоку **try** припиняється, і виконується відповідний блок **catch** або **finally**.
9. Розкрийте призначення та особливості роботи блоку catch.
Блок catch: Блок **catch** використовується для обробки виключень, які виникають у відповідному блоку **try**. У цьому блоку ви можете вказати, як обробляти виняток та виконати необхідні дії для відновлення.
10. Розкрийте призначення та особливості роботи блоку finally.
Блок finally: Блок **finally** використовується для виконання коду, який завжди має виконуватися, незалежно від того, чи виникали виключення в блоку **try** або **catch**. Це може використовуватися для звільнення ресурсів або завершення певних операцій.

Висновок: оволоділа навичками використання механізму виключень при написанні програм мовою Java.