Міністерство освіти і науки України
Національний університет «Львівська політехніка»

Кафедра «ЕОМ»

Звіт

до лабораторної роботи № 4

з дисципліни: «Кросплатформні засоби програмування»

На тему: «Виключення»

Виконав:
студент групи КІ-307
Возний А. О.

Перевірив:
доцент кафедри ЕОМ
Іванов Ю. С.

Львів – 2023

Мета роботи: 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.

Варіант завдання: y=cos(x)/sin(x)

Лістинг програми:

Клас Equations

```
package ki307.voznyi.lab4;

import java.io.BufferedWriter;
import java.io.FileWriter;
import java.io.IOException;
import java.io.PrintWriter;

/**
 * Клас Equations для обчислення та запису результату функції cos(x)/sin(x) у файл.
 */
public class Equations {
    private double x;
    private double x_rad;
    private PrintWriter fout;

    /**
     * Конструктор за замовчуванням, ініціалізує значення x та x_rad.
     */
    public Equations() {
        this.x = 0;
        this.x_rad = 0;
    }

    /**
     * Конструктор з параметром, ініціалізує значення x та x_rad зазначеним значенням
x у градусах.
     *
     * @param x Значення x у градусах.
     */
```

```
    public Equations(double x) {
        this.x = x;
        this.x_rad = x * 3.14159 / 180;
    }

    /**
     * Метод для обчислення виразу cos(x)/sin(x).
     *
     * @return Результат обчислення виразу cos(x)/sin(x).
     * @throws ArithmeticException Виняток, якщо sin(x) дорівнює 0.
     */
    public double Calc() throws ArithmeticException {
        if (Math.sin(x_rad) == 0) {
            throw new ArithmeticException("Exeption: sin(x) is equal to 0!\n");
        } else {
            return Math.cos(x_rad) / Math.sin(x_rad);
        }
    }

    /**
     * Метод для запису результату обчислення у файл "Lab4.txt".
     */
    public void in_file() {
        try {
            fout = new PrintWriter(new BufferedWriter(new FileWriter("Lab4.txt")));
            fout.print("cos(x)/sin(x) = " + Calc() + "\n");

        } catch (IOException | ArithmeticException e) {
            System.err.println("Can't use the file!!!\n");
        }
    }

    /**
     * Метод для закриття файлу після використання.
     */
    public void close_file() {
        fout.close();
    }
}
```

Клас EquationsApp

```
package ki307.voznyi.lab4;

import java.util.Scanner;

/**
 * Головний клас EquationsApp для введення значення x, обчислення та виведення результату функції cos(x)/sin(x) та запису його у файл "Lab4.txt".
 */
public class EquationsApp {

    /**
     * Точка входу в програму.
     * Вводить значення x, створює об'єкт класу Equations, обчислює та виводить результат функції cos(x)/sin(x) та записує його у файл "Lab4.txt".
     *
     * @param args Параметри командного рядка (не використовуються в даному випадку).
     */
    public static void main(String[] args) {
```

```java
        double x = 0;
        Scanner input = new Scanner(System.in);
        System.out.print("Input X: ");

        x = input.nextDouble();

        Equations eq = new Equations(x);

        System.out.print("cos(x)/sin(x) = " + eq.Calc() + "\n");

        eq.in_file();
        eq.close_file();
        input.close();

    }

}
```

Результат виконання програми:

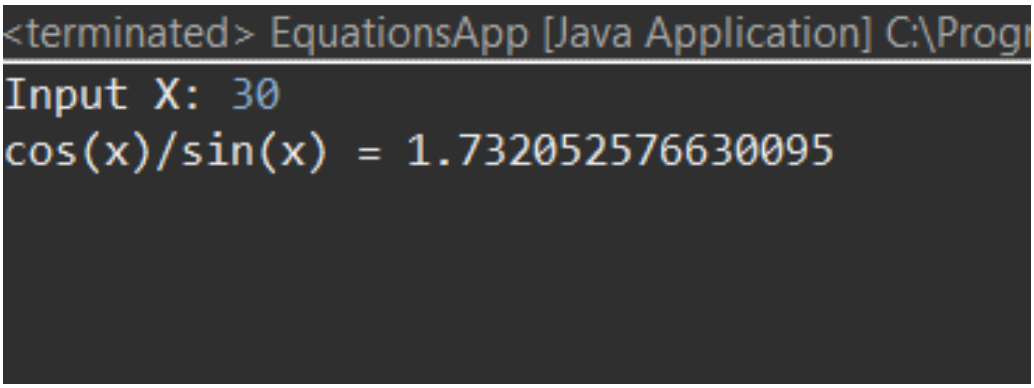

Рис. 1. Результат виконання програми

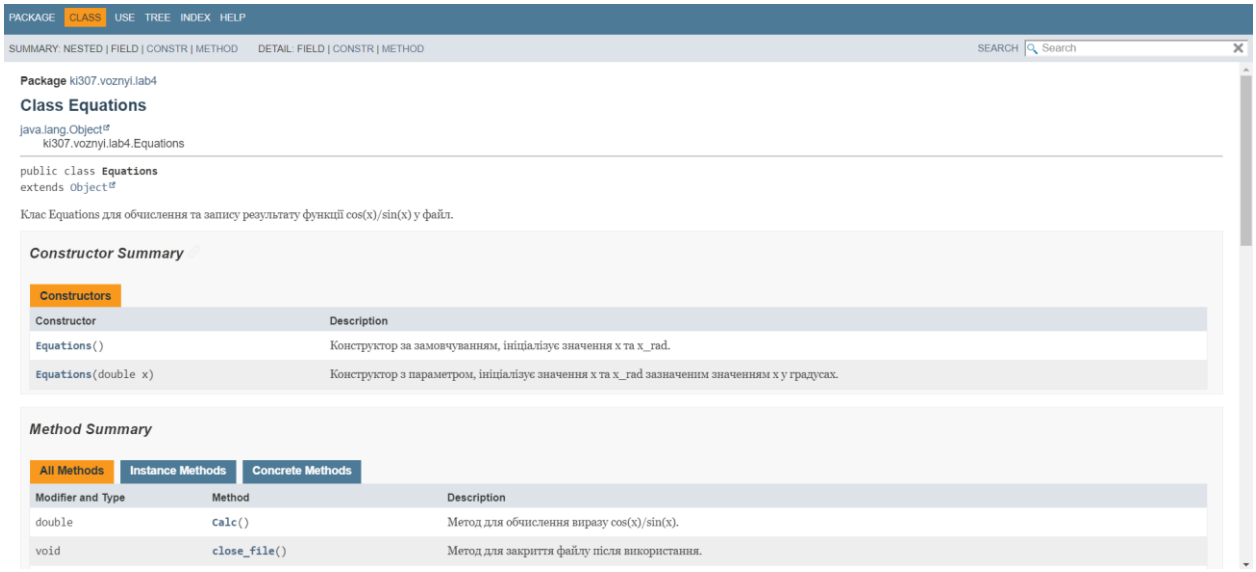

Рис. 2. Згенерована документація до класу Клас Equations

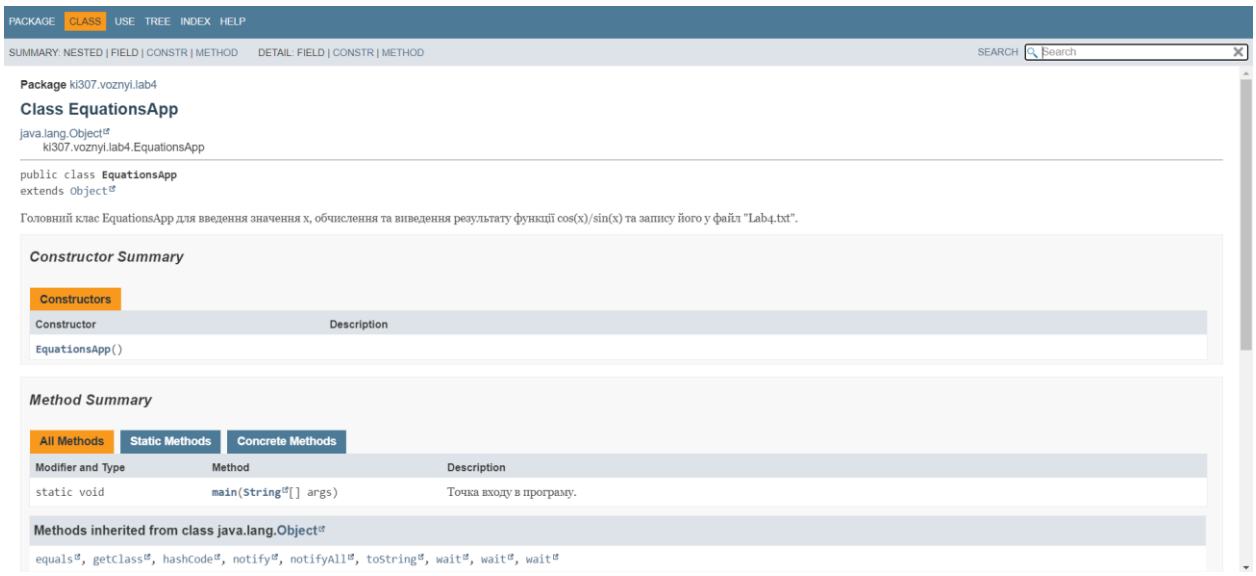

Рис. 3. Згенерована документація до класу EquationsApp

Відповіді 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: під час виконання цієї лабораторної роботи я оволодів навиками використання механізму виключень при написанні програм мовою Java.