Міністерство освіти і науки України НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»

Кафедра ЕОМ



Звіт

з лабораторної роботи № 4

з дисципліни «Кросплатформенні засоби програмування» на тему: «Виключення»

Виконав: ст. гр. КІ-304 Мох М. П.

Перевірив: викладач

Олексів М.В.

**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.Lab5 та володіти коментарями, які дозволять автоматично згенерувати документацію до розробленого пакету.
2. Автоматично згенерувати документацію до розробленого пакету.
3. Скласти звіт про виконану роботу з приведенням тексту програми, результату її виконання та фрагменту згенерованої документації.
4. Дати відповідь на контрольні запитання.

**Силка на GitHub:** https://github.com/MarianMoh/LabsMohKI-304

**Варіант 12:** y = sin(x) / tg(4x):

Клас Main:

package KI304.Moh.Lab4;  
  
import java.io.FileNotFoundException;  
import java.util.InputMismatchException;  
import java.util.Scanner;  
  
*/\*\*  
 \* The Main class contains the main method to execute the Calculator application.  
 \*  
 \* @author Marian Moh  
 \* @version 1.0  
 \* @since 2023-11-27  
 \*/*public class Main {  
  
 */\*\*  
 \* The main method of the application that takes user input, performs calculations,  
 \* and saves the result to a file using the Calculator class.  
 \*  
 \* @param args Command-line arguments (not used in this application).  
 \*/* public static void main(String[] args) {  
 System.*out*.println("Formula is: y = sin(x) / tg(4x)");  
 Scanner scanner = new Scanner(System.*in*);  
 while (true) {  
 System.*out*.print("\nEnter x: ");  
  
 try {  
 double x = scanner.nextDouble();  
 Calculator.*calculate*(x);  
 Calculator.*saveResultToFile*();  
 break;  
 } catch (InputMismatchException e) {  
 System.*out*.println("Input miscast, please enter a double");  
 scanner.next();  
 } catch (ArithmeticException e) {  
 System.*out*.println("Divide by 0!" + e.getMessage());  
 } catch(FileNotFoundException e) {  
 System.*out*.println("File not found... ");  
 break;  
 }  
 }  
 }  
}

# Клас Calculator:

package KI304.Moh.Lab4;  
  
import java.io.FileNotFoundException;  
import java.io.PrintWriter;  
  
*/\*\*  
 \* The Calculator class provides methods for mathematical calculations and result storage.  
 \* The results can be calculated using the formula y = sin(x) / tan(4x) and saved to a file.  
 \*  
 \* @author Marian Moh  
 \* @version 1.0  
 \* @since 2023-11-27  
 \*/*public class Calculator {  
 */\*\*  
 \* Holds the calculated result of the mathematical expression y = sin(x) / tan(4x).  
 \*/* private static double *result*;  
  
 */\*\*  
 \* The name of the file used to save the calculation results.  
 \*/* private static final String *FILE\_NAME* = "result.txt";  
  
 */\*\*  
 \* Calculates the result of the mathematical expression y = sin(x) / tan(4x).  
 \*  
 \* @param x The input value for the calculation.  
 \* @return The result of the mathematical expression.  
 \* @throws ArithmeticException If the denominator (Math.tan(4 \* x)) is 0, indicating division by zero.  
 \*/* public static double calculate(double x) {  
 if(Math.*tan*(4 \* x) == 0.0) {  
 throw new ArithmeticException("Math.tan(4 \* " + x + ") is 0, and we can't divide by it!");  
 }  
 *result* = Math.*sin*(x) / Math.*tan*(4 \* x);  
 System.*out*.println("Result: " + *result*);  
 return *result*;  
 }  
  
 */\*\*  
 \* Saves the result of the calculation to a file.  
 \*  
 \* @throws FileNotFoundException If the specified file for result storage is not found.  
 \*/* public static void saveResultToFile() throws FileNotFoundException {  
 try (PrintWriter printWriter = new PrintWriter(*FILE\_NAME*)) {  
 printWriter.println("Result y=sin(x)/tg(4x): " + *result*);  
 }  
 }  
}

# Результат роботи:

# 

# 

# Згенерована документація:

# 

**Відповіді на контрольні запитання:**

1. Дайте визначення терміну «виключення».

механізм мови Java, що забезпечує негайну передачу керування

блоку коду опрацювання критичних помилок при їх виникненні уникаючи процесу розкручування стеку

1. У яких ситуаціях використання виключень є виправданим?

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.

1. Яка ієрархія виключень використовується у мові Java?

Всі виключення в мові Java поділяються на контрольовані і неконтрольовані та спадкуються від суперкласу Throwable

1. Як створити власний клас виключень?

Для створення власного класу виключень в Java, спадкуйте ваш клас від одного з існуючих класів контрольованих виключень, додайте конструктори та використовуйте його для генерації виключень у вашому коді.

1. Який синтаксис оголошення методів, що можуть генерувати виключення?  
    public ReturnType methodName(Parameters) throws ExceptionType {

// Код методу

}

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

ті виключення, які можуть бути згенеровані з внутрішнього методу і які повинні оброблятися викликаючим кодом.

1. Як згенерувати виключення?

Генерація контрольованих виключень відбувається за допомогою ключового слова throw після якого необхідно вказати об’єкт класу виключення який і є власне виключенням, що генерує метод

1. Розкрийте призначення та особливості роботи блоку try.

Блок try використовується для обгортання коду, який може генерувати виключення. Він служить для відстеження виключень під час виконання коду в блоку.

1. Розкрийте призначення та особливості роботи блоку catch.

Блок catch використовується для обробки виключень, які були сгенеровані в блоку try. Може бути кілька блоків catch для обробки різних типів виключень.

1. Розкрийте призначення та особливості роботи блоку finally.

Блок finally використовується для виконання коду, який повинен виконатися завжди, незалежно від того, чи виникло виключення чи ні. Це корисно, наприклад, для звільнення ресурсів.

# Висновок:

У ході виконання даної лабораторної роботи, отримав навички використання механізму виключень при написанні програм мовою Java. Я вивчив, як обробляти винятки та використовувати блоки try, catch і finally для забезпечення безпеки та надійності мого коду.