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-301

*Солтисяк О. Р.*

**Прийняв:**

доцент кафедри ЕОМ

*Іванов Ю. С.*

Львів – 2024

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

**Завдання (варіант № 26)**

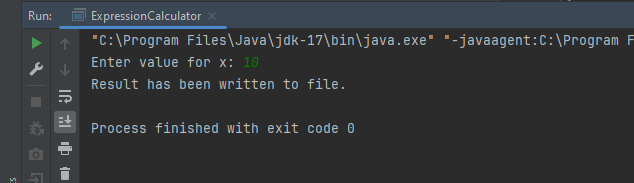
1. Створити клас, що реалізує метод обчислення виразу заданого варіантом **(**y=1/cos(4x)**)**. Написати на мові Java та налагодити програму-драйвер для розробленого класу. Результат обчислень записати у файл. При написанні програми застосувати механізм виключень для виправлення помилкових ситуацій, 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.

**Вихідний код програми:**

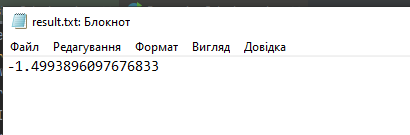
***Файл ExpressionCalculator.java***

//  
// Source code recreated from a .class file by IntelliJ IDEA  
// (powered by FernFlower decompiler)  
//  
  
package Lab4;  
  
import java.io.FileWriter;  
import java.io.IOException;  
import java.util.Scanner;  
  
public class ExpressionCalculator {  
 public ExpressionCalculator() {  
 }  
  
 public double calculateExpression(double x) throws ArithmeticException {  
 double denominator = Math.*cos*(4.0 \* x);  
 if (denominator == 0.0) {  
 throw new ArithmeticException("Cosine of 4x is zero, division by zero.");  
 } else {  
 return 1.0 / denominator;  
 }  
 }  
  
 public void writeToFile(double y, String filename) {  
 try {  
 FileWriter writer = new FileWriter(filename);  
  
 try {  
 writer.write(String.*valueOf*(y));  
 } catch (Throwable var8) {  
 try {  
 writer.close();  
 } catch (Throwable var7) {  
 var8.addSuppressed(var7);  
 }  
  
 throw var8;  
 }  
  
 writer.close();  
 } catch (IOException var9) {  
 IOException e = var9;  
 System.*err*.println("Error writing to file: " + e.getMessage());  
 e.printStackTrace();  
 }  
  
 }  
  
 public static void main(String[] args) {  
 ExpressionCalculator calculator = new ExpressionCalculator();  
  
 try {  
 Scanner scanner = new Scanner(System.*in*);  
  
 try {  
 System.*out*.print("Enter value for x: ");  
 double x = scanner.nextDouble();  
 double result = calculator.calculateExpression(x);  
 calculator.writeToFile(result, "result.txt");  
 System.*out*.println("Result has been written to file.");  
 } catch (Throwable var8) {  
 try {  
 scanner.close();  
 } catch (Throwable var7) {  
 var8.addSuppressed(var7);  
 }  
  
 throw var8;  
 }  
  
 scanner.close();  
 } catch (ArithmeticException var9) {  
 ArithmeticException e = var9;  
 System.*err*.println("Error: " + e.getMessage());  
 e.printStackTrace();  
 } catch (Exception var10) {  
 Exception e = var10;  
 System.*err*.println("Invalid input: " + e.getMessage());  
 e.printStackTrace();  
 }  
  
 }  
}

**Результат виконання програми:**



**Вміст файлу result.txt:**



**Відповіді на контрольні запитання:**

1. Дайте визначення терміну «виключення».

Виключення (або Exception) - це подія або об'єкт, що виникає під час виконання програми і може порушити нормальний хід виконання програми. Виключення вказують на помилку або несподівану ситуацію, яка вимагає обробки.

1. У яких ситуаціях використання виключень є виправданим?

Виключення слід використовувати, коли програма стикається з помилкою або непередбачуваною ситуацією, і ви хочете обробити цю ситуацію, щоб уникнути аварійного завершення програми. Це може включати в себе роботу з файлами, мережами, даними користувача тощо.

1. Яка ієрархія виключень використовується у мові Java?

У мові Java виключення представлені в ієрархічній структурі, починаючи з класу “java.lang.Throwable”. Ця ієрархія розділена на дві основні гілки: “java.lang.Error” (помилки, які не можна відновити) і “java.lang.Exception” (загальні виключення, які можна обробляти). Від “Exception” походять багато підкласів, наприклад, “RuntimeException”, “IOException”, “SQLException”, і т. д.

1. Як створити власний клас виключень?

Для створення власного класу виключень у Java вам потрібно створити клас, який успадковується від java.lang.Exception або від одного з його підкласів. Ваш клас повинен мати конструктор і може містити додаткові поля та методи, що допомагатимуть при обробці виключення.

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

public void myMethod() throws MyCustomException {

}

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

Виключення вказуються у заголовках методів за допомогою throws, якщо метод може викидати ці виключення. Це допомагає іншим програмістам розуміти, які виключення можуть виникнути при виклику методу і обробляти їх відповідно.

1. Як згенерувати контрольоване виключення?

throw new MyCustomException("Власне виключення");

1. Розкрийте призначення та особливості роботи блоку try.

Блок “try” використовується для обрамлення коду, в якому можуть виникати виключення. Якщо виключення виникає в блоці “try”, воно може бути оброблене в одному або кількох блоках “catch”. Якщо виключення не виникає, блок “catch” не виконується.

1. Розкрийте призначення та особливості роботи блоку catch.

Блок “catch” використовується для обробки виключень, які виникли в блоці “try”. Кожен блок “catch” вказує клас виключення, який він обробляє, і містить код обробки.

1. Розкрийте призначення та особливості роботи блоку finally.

Блок “finally” використовується для виконання коду, який завжди повинен виконуватися, незалежно від того, чи виникло виключення, чи ні. Це дозволяє вам виконати завершальні дії, такі як закриття ресурсів, навіть у випадку помилки.

**Висновок:**

На даній лабораторній роботі я отримав навички використання механізму виключень при написанні програм мовою Java.