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

Ширий Б. І.

Прийняв:

доцент кафедри ЕОМ

Іванов Ю. С.

Львів – 2023

МЕТОДИЧНІ ВІДОМОСТІ РОБОТИ

МЕТА

Оволодіти навиками використання механізму виключень при написанні програм мовою Java.

ЗАВДАННЯ

№1

Створити клас, що реалізує метод обчислення виразу заданого варіантом, а саме

$$y = \frac{1}{\cos 4x}.$$

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

Для розробленої програми згенерувати документацію

№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.

ВИКОНАННЯ ЛАБОРАТОРНОЇ РОБОТИ

ВИХІДНИЙ КОД

Написав програму, що реалізує метод обчислення виразу $y = \frac{1}{\cos}$, код якої наведено у лістингу 2.1.

Лістинг 2.1. Код основної програми.

```
package CI_306.Shyryi.Lab4;

import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.io.FileWriter;
import java.io.IOException;
import java.io.PrintWriter;

/**
 * The ExpressionCalculatorGUI class provides a graphical user interface for
 * performing mathematical calculations based on user input. It allows the user
 * to input a value for 'x', select whether to use the value of π (pi), and specify
 * a file name for saving the calculation result.
 */
public class ExpressionCalculatorGUI {
         private JTextField xTextField;
    private JTextField fileNameTextField;
    private JButton calculateButton;
    private JButton openFileButton;
    private JTextArea resultTextArea;
    JCheckBox checkBox = new JCheckBox("пі");
    /**
     * Constructs the ExpressionCalculatorGUI, creating the graphical user interface
     * and setting up event listeners for the calculate and open file buttons.
     */
    public ExpressionCalculatorGUI() {
        JFrame frame = new JFrame("Calculator");
        frame.setSize(600, 150);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setLayout(new BorderLayout());
        JPanel inputPanel = new JPanel(new GridLayout(2, 4, 0, 20));
        JLabel xLabel = new JLabel("Введіть значення x:");
        xTextField = new JTextField();
        checkBox.setSize(1,1);
        JLabel fileNameLabel = new JLabel("Введіть назву файлу:");
        fileNameTextField = new JTextField();
        calculateButton = new JButton("Обчислити");
        openFileButton = new JButton("Відкрити файл");
        inputPanel.add(xLabel);
        inputPanel.add(xTextField);
        inputPanel.add(calculateButton);
        inputPanel.add(checkBox);
        inputPanel.add(fileNameLabel);
        inputPanel.add(fileNameTextField);
        inputPanel.add(openFileButton);

        resultTextArea = new JTextArea();
        resultTextArea.setEditable(false);

        frame.add(inputPanel, BorderLayout.NORTH);
        frame.add(new JScrollPane(resultTextArea), BorderLayout.CENTER);

        calculateButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                calculateExpression();
            }
        });

        openFileButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
```

```
                openFile();
            }
        });

        frame.setVisible(true);
    }

    /**
     * Calculates the mathematical expression based on user input for 'x' and the
     * selection of π (pi). Saves the result to a specified file or a default file
     * if no file name is provided.
     */
    private void calculateExpression() {
        try {
            double x = Double.parseDouble(xTextField.getText());
            double result = 1 / ((checkBox.isSelected() && ((Math.abs(4 * x) == 0.5) ||
                                                    (((4 * x - 0.5) % 1) == 0))) ?
                                                    0 : Math.cos(4 * x));

                            if (result == Double.NaN
                                    || result == Double.NEGATIVE_INFINITY
                                    || result == Double.POSITIVE_INFINITY)
                                    { throw new ArithmeticException(); }


            String fileName = fileNameTextField.getText();
            if (fileName.isEmpty()) {
                fileName = "result.txt";
            }
            PrintWriter writer = new PrintWriter(new FileWriter(fileName + ".txt"));
            writer.println("Значення у при x = " + x + " дорівнює " + result);
            writer.close();
            resultTextArea.setText("Результат обчислення записано у файл '"+ fileName+ "'");


        } catch (NumberFormatException e) {
            resultTextArea.setText("Помилка: Невірний формат введених даних для x");
        } catch (ArithmeticException e) {
         resultTextArea.setText("Ділення на нуль: cos(4x) дорівнює нулю.");
        } catch (IOException e) {
            resultTextArea.setText("Помилка запису в файл: " + e.getMessage());
        }
    }

    /**
     * Opens a text file with the specified file name or displays an error message
     * if the file cannot be opened.
     */
    private void openFile() {
        String fileName = fileNameTextField.getText();
        try {
            java.awt.Desktop.getDesktop().open(new java.io.File(fileName + ".txt"));
        } catch (IOException e) {
            resultTextArea.setText("Помилка відкриття файлу: " + e.getMessage());
        } catch (IllegalArgumentException e) {
            resultTextArea.setText("Помилка: " + e.getMessage());
        }
    }

    /**
     * The main method that creates an instance of ExpressionCalculatorGUI and
     * initializes the GUI within the Swing event dispatch thread.
     *
     * @param args The command-line arguments (not used in this application).
     */
    public static void main(String[] args) {
        SwingUtilities.invokeLater(new Runnable() {
            @Override
            public void run() {
                new ExpressionCalculatorGUI();
            }
        });
    }
}
```

РЕЗУЛЬТАТИ ВИКОНАННЯ

Початковий вигляд програми наведено на рисунку 2.1.

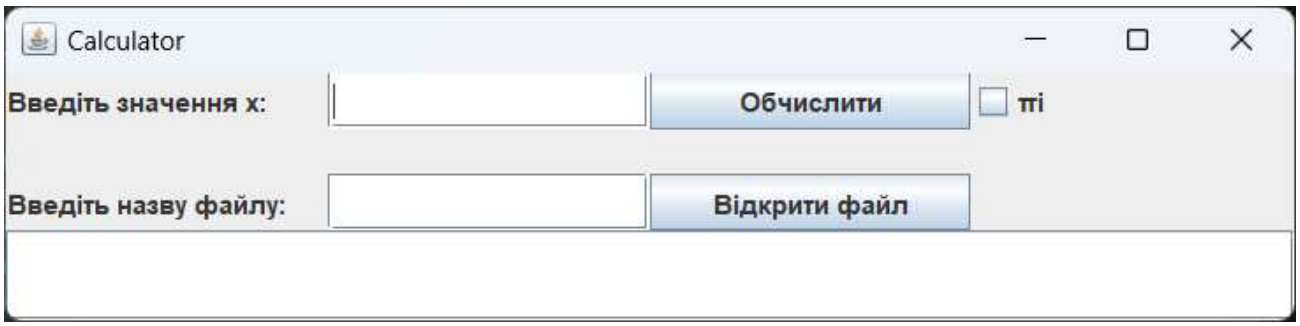

Рисунок 2.1. Початковий вигляд програми.

Згідно завдання передбачаю багато виключень, такі як:

- Введення неправильного формату даних - рисунок 2.2,
- Ділення на нуль - рисунок 2.3,
- Створення файлу з некоректною назвою - рисунок 2.4,
- Читання неіснуючого файлу - рисунок 2.5.

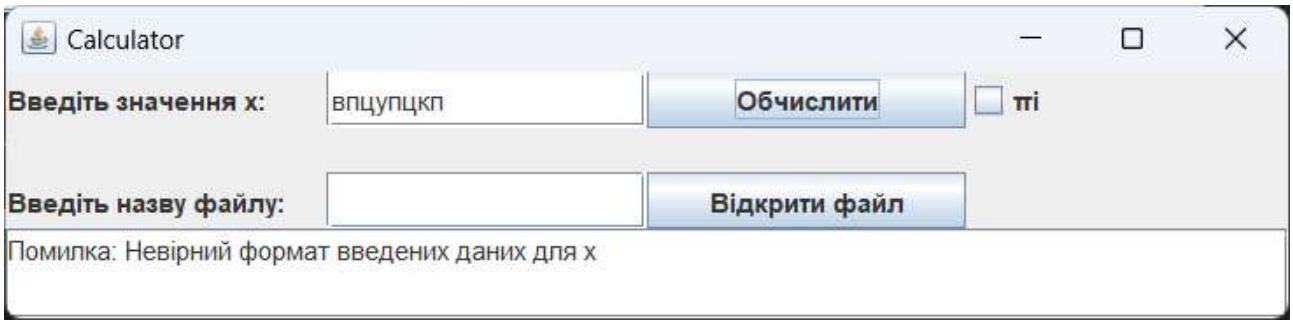

Рисунок 2.2. Помилка формату даних.

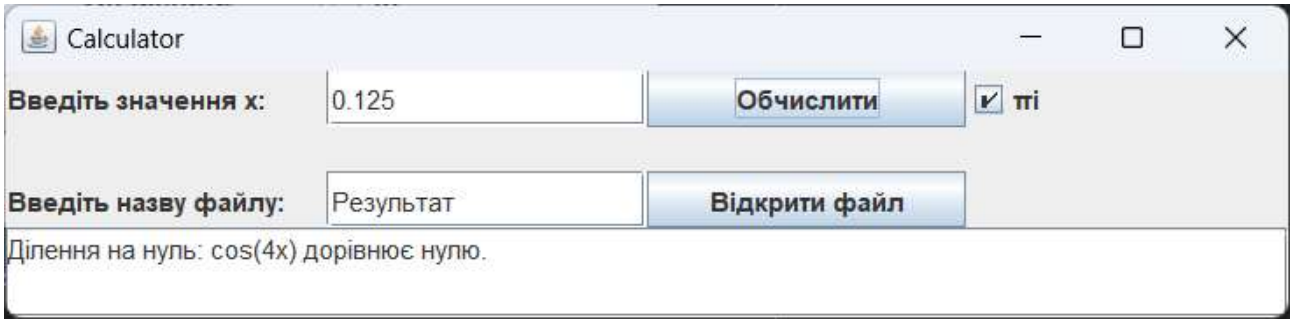

Рисунок 2.3. Помилка ділення на нуль.

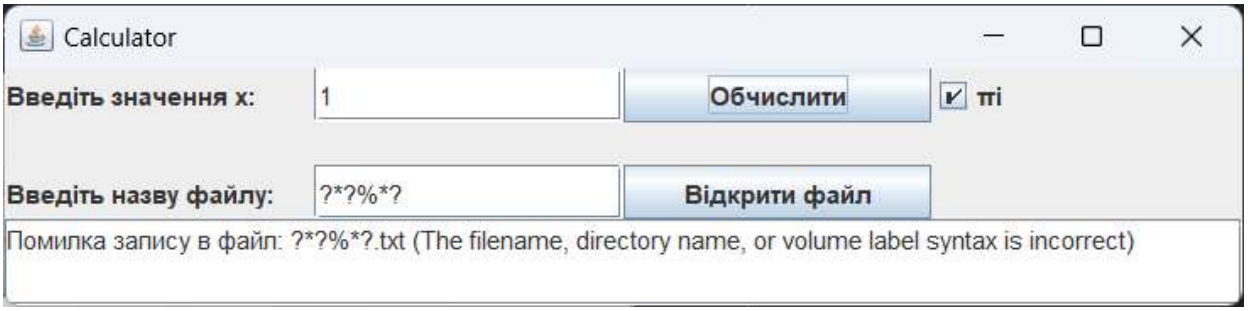

Рисунок 2.4. Помилка некоректної назви файлу.

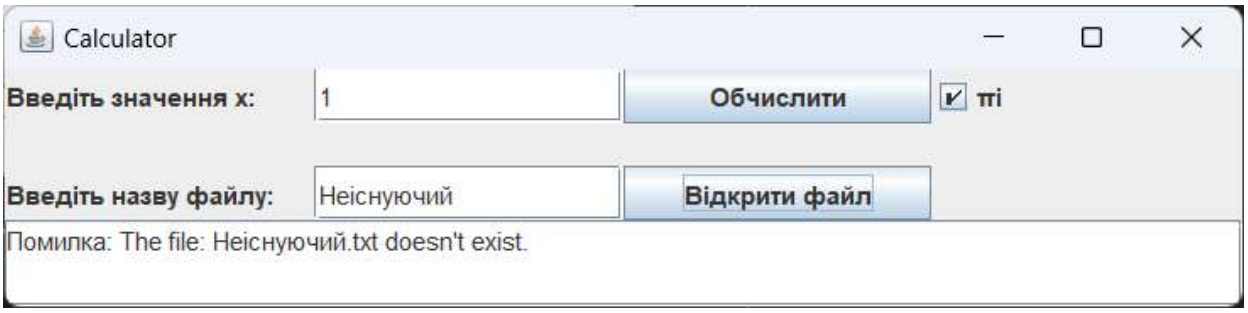

Рисунок 2.5. Помилка відкриття неіснуючого файлу.

Якщо всі файли введені коректно, то ми отримаємо відповідь записану у файлі.

На рисунку 2.6 наведена поведінка програми, а на рисунку 2.7 файл зі записом відповіді.

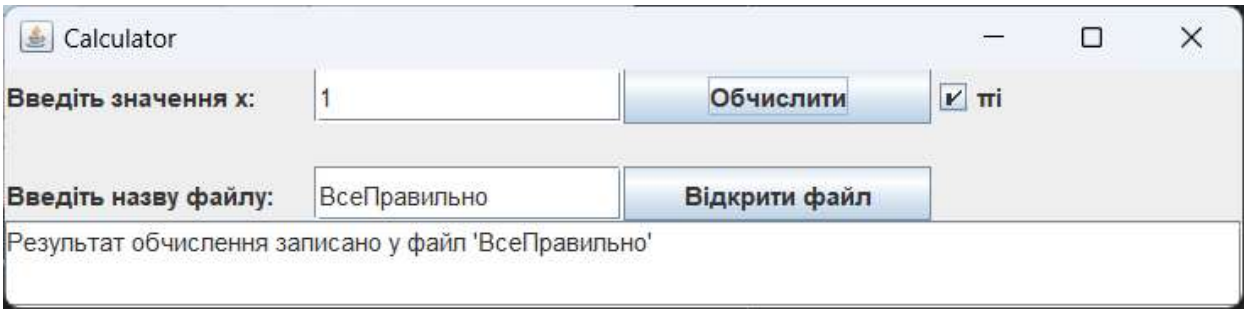

Рисунок 2.6. Правильна поведінка з програмою.

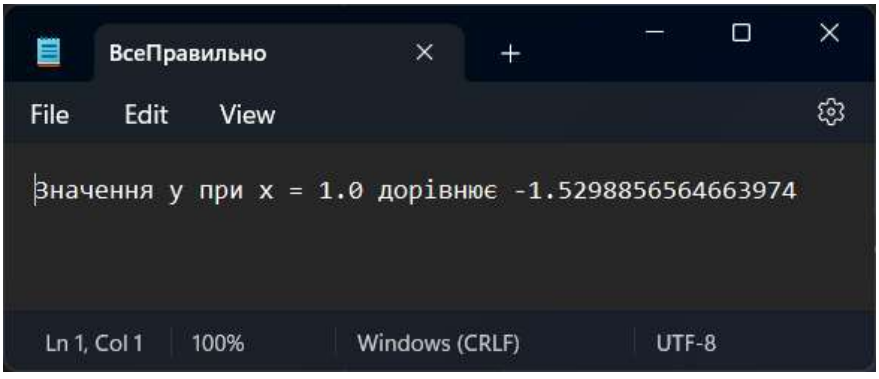

Рисунок 2.7. Запис у програмі при коректній поведінці користувача.

ДОКУМЕНТАЦІЯ

Згенерував документацію, фрагмент якої навів на рисунку 2.8.

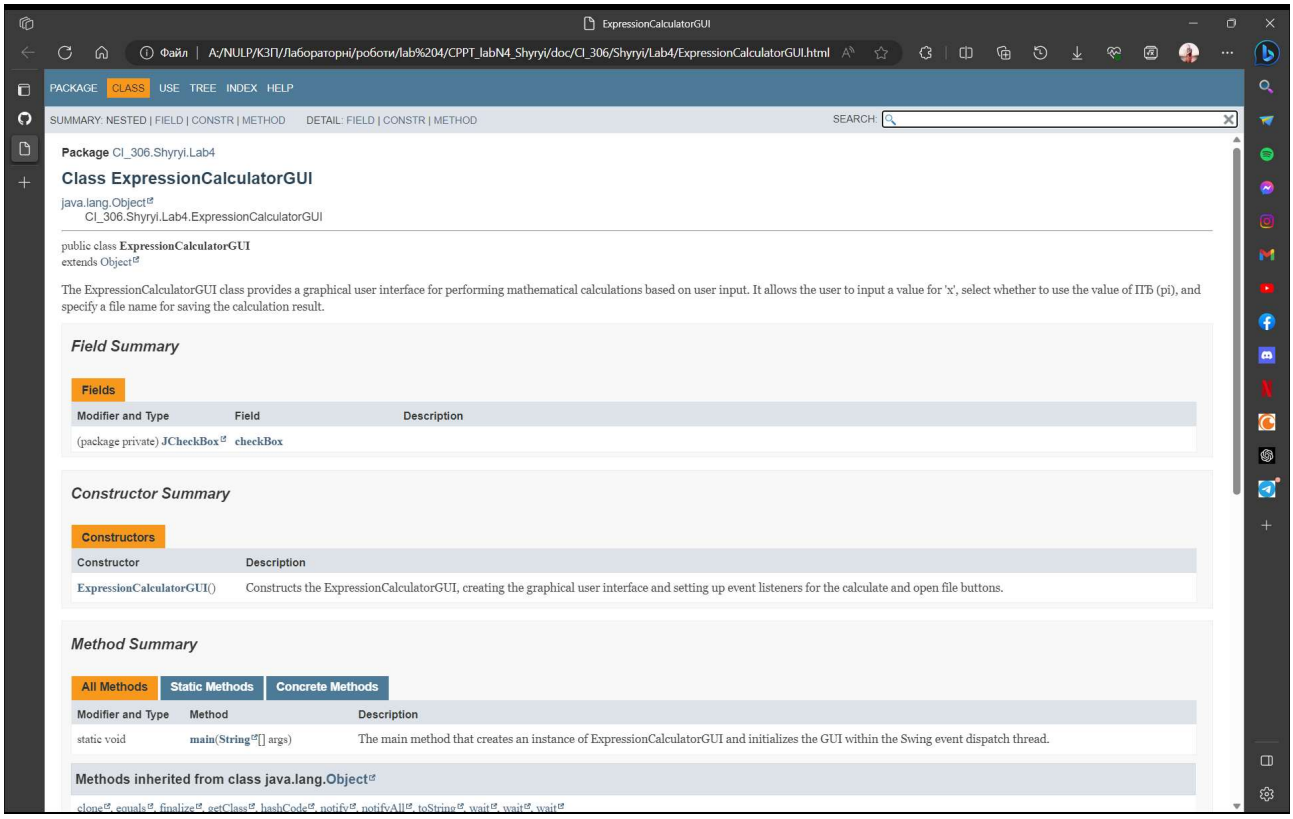

Рисунок 2.8. Фрагмент згенерованої документації.

ВІДПОВІДІ НА КОНТРОЛЬНІ ПИТАННЯ

ДАЙТЕ ВИЗНАЧЕННЯ ТЕРМІНУ «ВИКЛЮЧЕННЯ».

Виключення - це негативна подія або помилкова ситуація, яка виникає під час виконання програми і порушує звичайний хід програми.

У ЯКИХ СИТУАЦІЯХ ВИКОРИСТАННЯ ВИКЛЮЧЕНЬ Є ВИПРАВДАНИМ?

Виключення виправдано використовувати, коли в програмі можуть виникнути непередбачені або помилкові ситуації, і потрібно забезпечити обробку цих ситуацій для збереження стабільності та безпеки програми.

ЯКА ІЄРАРХІЯ ВИКЛЮЧЕНЬ ВИКОРИСТОВУЄТЬСЯ У МОВІ JAVA?

Ієрархія виключень включає клас Throwable як корінь, з двома підкласами: Exception і Error. Exception поділяється на контрольовані (checked) і неконтрольовані (unchecked) виключення.

ЯК СТВОРИТИ ВЛАСНИЙ КЛАС ВИКЛЮЧЕНЬ?

Для створення власного класу виключень потрібно створити підклас від класу Exception або його підкласу.

ЯКИЙ СИНТАКСИС ОГОЛОШЕННЯ МЕТОДІВ, ЩО МОЖУТЬ ГЕНЕРУВАТИ ВИКЛЮЧЕННЯ?

Методи, що можуть генерувати виключення, оголошуються з ключовим словом "throws" після списку параметрів методу.

ЯКІ ВИКЛЮЧЕННЯ СЛІД ВКАЗУВАТИ У ЗАГОЛОВКАХ МЕТОДІВ І КОЛИ?

Слід вказувати контрольовані (checked) виключення, які можуть виникнути в результаті дій користувача або зовнішніх факторів, а також всі виключення підкласу RuntimeException.

ЯК ЗГЕНЕРУВАТИ КОНТРОЛЬОВАНЕ ВИКЛЮЧЕННЯ?

Використовуйте ключове слово "throw" для створення екземпляру власного класу виключення та його викидання.

РОЗКРИЙТЕ ПРИЗНАЧЕННЯ ТА ОСОБЛИВОСТІ РОБОТИ БЛОКУ TRY.

Блок try використовується для обрамлення коду, який може викликати виключення. Всі виключення, які виникають у цьому блоку, перехоплюються і передаються для обробки в блоки catch або finally.

РОЗКРИЙТЕ ПРИЗНАЧЕННЯ ТА ОСОБЛИВОСТІ РОБОТИ БЛОКУ CATCH.

Блок catch використовується для обробки і відловлювання виключень, які виникли у блоку try. Він містить код для обробки конкретного типу виключення.

РОЗКРИЙТЕ ПРИЗНАЧЕННЯ ТА ОСОБЛИВОСТІ РОБОТИ БЛОКУ FINALLY.

Блок finally використовується для виконання коду, незалежно від того, чи виникли виключення в блоку try. Він виконується завжди, навіть якщо виключення було перехоплено або не було викинуто.

ВИСНОВОК

У цій лабораторній роботі був створений клас для обчислення виразу заданого варіантом і написано програму-драйвер для цього класу. Механізм виключень використовувався для виправлення можливих помилкових ситуацій у виконанні програми. Програма розміщена в пакеті CI_306.Shyryi.Lab5 і містить коментарі, що дозволили згенерувати документацію до пакету. Документація була згенерована автоматично. У звіті були представлені текст програми, результати її виконання та фрагменти згенерованої документації.