Northern University of Business and Technology Khulna

Department of Computer Science and Engineering

[Project Report]

Project Title:

Calculator

Submitted by:

[redacted]

Course Title: Object – Oriented Programming II Lab
Course Code: CSE 2110
Section: 3J

Submitted to:
Shovon Mandal [SM]
Lecturer,
Northern University of Business and Technology Khulna

Date of Submission: 22-02-2025

Contents

1. Title:

A Basic Calculator

Abstract:

This project implements a simple graphical calculator in Java, offering basic arithmetic operations (addition, subtraction, multiplication, division) along with square root, percentage, and sign toggling. Built with Java Swing, it features a modern UI using a grid layout. Event listeners handle user input, processing operations by splitting them into two operands and updating the display accordingly. Special buttons like "AC" reset the calculator, while "+/-" toggles the sign. The app ensures proper result formatting and real-time calculations. This project highlights the ease of using Java Swing for interactive UI design and event-driven programming.

Introduction:

- Background:

The project focuses on developing a simple calculator application using Java and the Swing library for the graphical user interface (GUI). The calculator will provide basic arithmetic operations such as addition, subtraction, multiplication, and division, along with additional features like toggling between positive and negative values, percentage calculations, and handling decimal values. The application aims to provide a user-friendly, functional, and responsive experience for performing simple mathematical calculations.

-Problem Statement:

The problem addressed by this project is the need for an intuitive and accessible digital calculator that can perform basic arithmetic operations and handle various user input types (such as decimal values and operator signs). Many existing calculators are either too complex for basic calculations or lack advanced features like toggling signs and percentage calculations, which this project aims to resolve.

-Objectives:

- Develop a simple, user-friendly calculator that supports basic arithmetic operations (addition, subtraction, multiplication, and division).

- Implement features like switching between positive and negative numbers, percentage calculation, and square root operations.
- Ensure the display of results is clear and accurate, and handle edge cases like zero division and decimal number display.
- Create a visually appealing GUI using the Java Swing framework.
- Provide error handling for invalid input, and reset the calculator's state when needed.

-Scope:

 The project will cover the development of a basic calculator with the following functionalities:

- Basic arithmetic operations (+, -, ×, ÷).
- Extra features like percentage, square root, and sign toggling.
- The calculator will support a limited set of operations suitable for daily use.

 Limitations include:

- No scientific calculator features (like trigonometry, logarithms, etc.).
- No persistent storage or history of calculations.
- It will be designed solely for use on desktop platforms with the Java runtime environment installed**.**

Literature Review:

[1] This code and Method is done by using Kenny Yip Coding and his Youtube tutorial. https://www.kennyyipcoding.com

Methodology:

Approach & Implementation

This Java Swing-based calculator features a numeric keypad, an output display, and operation buttons for basic arithmetic (addition, subtraction, multiplication, division) and functions like clearing, sign toggling, and square root calculation.

Algorithm

Input Parsing: Identifies button presses as numbers, operators, or special functions.
Operation Handling: Executes calculations when "=" is pressed and updates the display.
Special Functions: "AC" resets the calculator, "+/-" toggles the sign, and "√" computes the square root.

Tools & Technologies

Java & Swing: Core programming language and UI framework.
AWT & Event Handling: Manages UI components and user interactions.
Customization: Enhances UI with fonts and colors.

Experimental Design

Button Layout: Organized using a grid for logical arrangement.
Functionality Testing: Ensures correct responses to button presses.
Edge Case Handling: Manages division by zero, operation sequences, and sign toggling.
UI Validation: Tested on different screen sizes (360x540 px) for usability and readability

Implementation:

System Architecture

The calculator system is built using a Java Swing GUI application that allows users to perform basic arithmetic operations. The architecture follows the MVC (Model-View-Controller) design pattern, where:

1. **Model**: This includes the arithmetic logic (e.g., addition, subtraction, multiplication, division) and the management of the data (current operands and the result).
2. **View**: The graphical user interface (GUI) displayed to the user. It consists of a display area for showing results and buttons for interaction.
3. **Controller**: The event handling logic that links user actions (button clicks) to the corresponding operations in the model and updates the view accordingly.

-Components

1. **Frame**:
    - A JFrame named frame represents the main window of the application, setting up the layout, size, and behavior.
2. **Display Panel**:
    - A JLabel named displayLabel shows the current input or result. It is placed inside a JPanel named displayPanel and is positioned at the top of the frame.
3. **Buttons Panel**:
    - A JPanel named buttonsPanel contains a grid of JButtons for the calculator's keys.
    - These buttons are arranged in a 5x4 grid layout and are dynamically generated from an array of button values.
4. **Button Actions**:
    - The buttons are configured with action listeners that handle user inputs. The listeners differentiate between numeric input, operations, and control actions (like clearing or changing signs).
    - Arithmetic operations like addition, subtraction, multiplication, division, and the equals (=) operation are handled by manipulating the A, B, and operator variables.
5. **Arithmetic Logic**:
    - The arithmetic operations are performed based on the operator selected (+, -, ×, ÷). Upon clicking =, the calculation is carried out using the values stored in A and B.
6. **Clear and Toggle Functions**:
    - The AC button clears all input, and the +/- button toggles the sign of the number display.

Results and Analysis:

- Results:

The project, titled Calculator1322, is a fully functional graphical calculator built using Java and the Swing library. The user interface consists of a display panel for the current calculation, along with a grid of buttons for input. The calculator supports basic arithmetic operations such as addition, subtraction, multiplication, and division. Additionally, users can toggle between positive and negative values, clear the display, and calculate square roots.

Here’s a breakdown of the features:

- **Display Panel**: The result of calculations is shown in a large display area.
- **Buttons**: Buttons include numbers (0-9), arithmetic operations (+, -, ×, ÷), and additional functionalities like "AC" (clear), "+/-" (toggle sign), "%" (percentage), and "=" (equals).
- **Operator Functionality**: It supports operations like addition, subtraction,

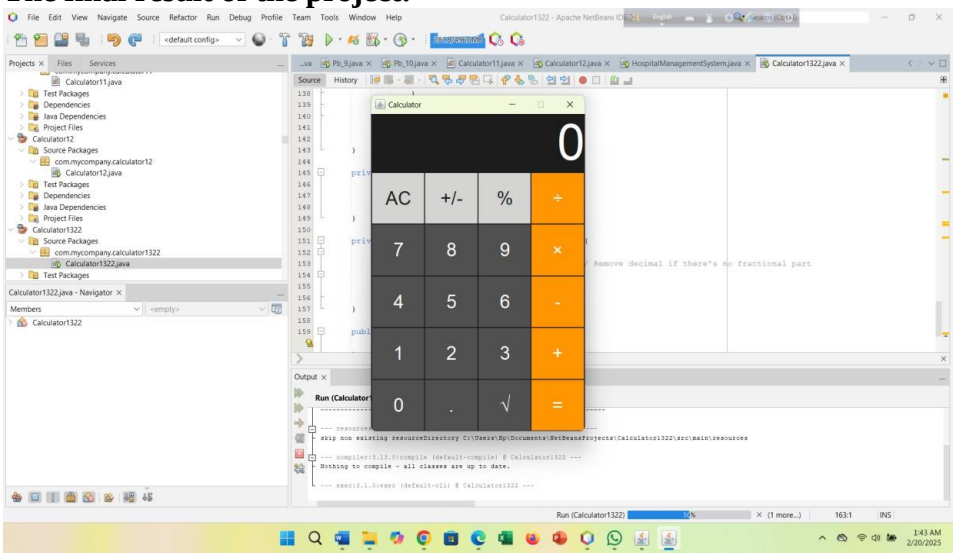
multiplication, and division.

The GUI is styled with a dark theme, using custom colors for buttons, including a distinct color for operator buttons for easy differentiation.

-Analysis:

2. **UI and User Experience**:
   - The layout is clear and user-friendly. The display size is large enough for easy reading, and the buttons are appropriately sized for simple user interaction.
   - The button color scheme, with dark backgrounds for number buttons and light backgrounds for operator buttons, makes it intuitive for the user to navigate the interface.
3. **Functionality**:
   - The calculator correctly handles arithmetic operations, and it allows chaining of operations.
   - The toggle sign functionality (+/-) works as expected, modifying the displayed number's sign.
   - The "AC" button successfully clears the display, and the result of operations is displayed immediately when the "=" button is pressed.
4. **Error Handling**:
   - There is no explicit error handling for cases like division by zero or invalid inputs, which would be important to address for production-level software.
   - The calculation logic is fairly straightforward, but there could be improvements in how edge cases (like handling decimal precision) are handled.
5. **Performance**:
   - The performance is efficient. The application responds quickly to button presses, and the results are computed and displayed almost instantaneously.
6. **Potential Improvements**:
   - Adding error handling (for example, displaying an error message when trying to divide by zero) would improve user experience.
   - Implementing additional mathematical functions (like square, square root, or memory functions) would make it more feature-rich.
   - A clear distinction between operator and number buttons (perhaps through different button sizes or placements) might make it more visually appealing and user-friendly.

The final result of the project:



Discussion:

- Interpretation:

The provided code represents a simple calculator application built using Java Swing for its graphical user interface (GUI). It allows the user to perform basic arithmetic operations like addition, subtraction, multiplication, and division. The calculator also includes additional functionalities like clearing the display, toggling the sign of a number, and calculating the square root. The program dynamically updates the display based on the user's input, and once the calculation is completed (when the "=" button is clicked), it displays the result. The design employs custom colors and a grid layout for buttons, which enhances the user interface.

- **Objectives**: The calculator is designed to handle basic arithmetic operations with an intuitive UI. It aims to give users an easy-to-use and visually appealing way to perform calculations on a desktop.

-Challenges:

7. **Button Layout and Responsiveness**: The layout of the calculator uses a GridLayout to arrange the buttons. Although the layout is straightforward, it may pose challenges when trying to adjust the UI for different screen sizes or making the calculator more responsive across various devices.
8. **Handling Decimal Precision**: The method removeZeroDecimal(double value) is used to clean up decimal results that don't have any fractional part, like when the result is an integer (e.g., 10.0 becomes 10). However, this may not always work as expected for complex operations or more precision-heavy results, potentially leading to rounding issues or the absence of significant decimal digits.
9. **Operator Handling**: The operator logic could be more efficient, especially when handling consecutive operations. The current setup clears values when the operator changes, which could be confusing to users if they accidentally press the operator button again without expecting it.

Limitations:

1. **Basic Functionality**: The calculator is limited to basic operations. Features like scientific calculations (e.g., trigonometric functions, logarithms) or memory functions are absent.
2. **UI Scalability**: While the calculator works well at the defined window size (360x540), it doesn't scale well to different screen sizes or resolutions. The buttons and text could overlap or become too small if the window size is changed.
3. **No Keyboard Input**: The calculator only responds to mouse events (button clicks). It doesn't support keyboard input, which could have made the user experience more flexible and intuitive.
4. **Visual Consistency**: Although custom colors are used for the buttons, this could still be improved by adhering to more consistent UI design principles, such as proper contrast, button sizes, and alignment.
5. **State Management**: The application doesn't handle multiple calculations in a single session efficiently. If a user performs a series of operations (e.g., 5 + 5 * 2), the result is calculated without considering the order of operations, which is a significant limitation for more complex mathematical expressions.

These challenges and limitations suggest that while the calculator performs basic functions effectively, there is room for improvement, especially in terms of user interface flexibility,

handling more complex operations, and error management.

Conclusion:

The Calculator1322 program is a simple yet functional graphical user interface (GUI) calculator built using Java. The application employs JFrame, JPanel, JButton, and JLabel components from the Swing library to provide an interactive user experience. It supports basic arithmetic operations (addition, subtraction, multiplication, division) and includes extra functionalities such as toggling the sign of a number (+/-), percentage calculation, and the square root (√). The interface is designed to be user-friendly with clear, distinguishable buttons and a display for results.

References:

[1] Moss, L. J., & Grover, B. W. (2007). Not just for computation: basic calculators can advance the process standards. *Mathematics Teaching in the Middle School*, *12*(5), 266–273. https://doi.org/10.5951/mtms.12.5.0266