CREATING GUI CALCULATOR USING PYTHON

1. Introduction:

The "Simple Calculator using Tkinter in Python" is a graphical user interface (GUI) application designed to perform basic arithmetic calculations with ease. Developed using the Tkinter library in Python, this calculator provides a user-friendly platform for numerical operations. The project offers a clean and intuitive design, featuring digit buttons, arithmetic operators, and special functions like square and square root. With a focus on simplicity and functionality, the calculator incorporates error handling for a seamless user experience. The graphical interface includes two distinct labels to display the ongoing expression and the cumulative expression history. By leveraging Tkinter's capabilities, this project serves as an excellent introduction to GUI development in Python, showcasing the integration of buttons, labels, and event handling to create an interactive and responsive calculator application.

1. Features:

* The calculator supports basic arithmetic operations: addition, subtraction, multiplication, and division.
* It has buttons for digits 0-9, decimal point, and clear.
* Special functions include square (x²), square root (√x), and equals (=) for calculation.
* The interface has a display for both the current expression being entered and the total expression history.
* Error handling is implemented to catch and display errors during evaluation.

1. GUI Design:

The GUI is designed using Tkinter frames, labels, and buttons.

Different fonts and colors are used to enhance the visual appeal of the calculator.

1. Code Structure:

**Initialization**:

* The `Calculator` class is initialized with necessary parameters and default expressions.
* GUI window properties such as size, title, and resizable are set.

**Display Labels:**

Two labels are created to display the current expression and the total expression history.

**Digit and Operator Buttons:**

* Digit buttons (0-9) and decimal point are created using a loop, and their positions are set on the grid.
* Operator buttons (+, -, \*, /) are created similarly.
* Digit and operator buttons are bound to functions for updating the expressions.

**Special Buttons:**

* Clear button (C) clears both current and total expressions.
* Square (x²) and square root (√x) buttons are implemented to square and take the square root of the current expression.

**Evaluation and Equals Button:**

* The `evaluate` function combines the total expression and current expression for calculation.
* The equals (=) button triggers the evaluation and updates the display.

**Key Bindings:**

The calculator supports key bindings for digit keys, Enter key (for equals), and operator keys.

**Update Functions:**

* ‘update\_total\_label` updates the total expression label with proper symbols for operators.
* `update\_label` updates the current expression label.

**Run Function:**

The `run` function starts the Tkinter main loop.

1. Conclusion:

* The project successfully creates a functional calculator GUI using Tkinter.
* It includes features like basic arithmetic operations, special functions, and error handling.
* The code is structured and easy to understand, making it suitable for beginners learning GUI development in Python.

1. **Future Work:**

Future iterations of the project could include a more sophisticated GUI design or integration with other Python libraries for enhanced functionality.