

1. **Operations:** The calculator should support addition, subtraction, multiplication, division, and potentially other operations.
2. **Input Prohibition:** The calculator should not accept arithmetic expressions (e.g., "15+10*8"). It should only allow a single operation at a time.
3. **GUI:** A graphical user interface (GUI) is preferred, and the program can use Tkinter, which is a simple and common library for GUIs in Python.
4. **Code Quality:** The code should be well-organized and readable.

Calculator with GUI using Tkinter

Given the requirements, I'll create a calculator application using Tkinter:

1. **Import the required libraries:** We need to import `tkinter` for creating the GUI.
2. **Create the calculator class:** This class will handle the GUI and operations of the calculator.
3. **Initialize the GUI:** We set up the calculator window, buttons, and display.
4. **Define operations:** The calculator will handle addition, subtraction, multiplication, and division.
5. **Implement button callbacks:** Define functions for button presses to update the display and perform operations.
6. **Run the application:** Start the main loop of the application.

```
import tkinter as tk
```

```
class Calculator:
```

```
    def __init__(self, root):  
        self.root = root  
        self.root.title("Calculator")
```

```
        # Create a display for the calculator  
        self.display = tk.Entry(root, font=("Arial", 16), justify='right', bd=10,  
insertwidth=4, width=14, borderwidth=4)  
        self.display.grid(row=0, column=0, columnspan=4)
```

```
        # Variable to hold the first operand and operation  
        self.first_operand = None  
        self.operation = None
```

```
        # Create buttons for the calculator  
        button_texts = [  
            '7', '8', '9', '/',  
            '4', '5', '6', '*',  
            '1', '2', '3', '-',
```

```

        'C', '0', '=', '+'
    ]

    # Create and place buttons in a 4x4 grid
    button_index = 0
    for i in range(4):
        for j in range(4):
            button_text = button_texts[button_index]
            button_index += 1
            button = tk.Button(root, text=button_text, padx=20, pady=20,
font=("Arial", 14),
                                command=lambda text=button_text: self.on_button_click(text))
            button.grid(row=i + 1, column=j)

def on_button_click(self, button_text):
    # Handle button clicks
    if button_text == 'C':
        # Clear the display
        self.display.delete(0, tk.END)
        self.first_operand = None
        self.operation = None
    elif button_text in ['+', '-', '*', '/']:
        # Store the first operand and operation
        try:
            self.first_operand = float(self.display.get())
            self.operation = button_text
            self.display.delete(0, tk.END)
        except ValueError:
            self.display.delete(0, tk.END)
            self.display.insert(tk.END, "Error")
    elif button_text == '=':
        # Perform the operation and display the result
        try:
            second_operand = float(self.display.get())
            result = self.calculate(self.first_operand, second_operand, self.operation)
            self.display.delete(0, tk.END)
            self.display.insert(tk.END, str(result))
            # Reset for next calculation
            self.first_operand = None
            self.operation = None
        except (ValueError, ZeroDivisionError):
            self.display.delete(0, tk.END)
            self.display.insert(tk.END, "Error")

```

```

else:
    # Append the button text (digit or point) to the display
    self.display.insert(tk.END, button_text)

def calculate(self, operand1, operand2, operation):
    # Perform the calculation based on the operation
    if operation == '+':
        return operand1 + operand2
    elif operation == '-':
        return operand1 - operand2
    elif operation == '*':
        return operand1 * operand2
    elif operation == '/':
        if operand2 == 0:
            raise ZeroDivisionError("Cannot divide by zero")
        return operand1 / operand2

# Create the main application window
root = tk.Tk()

# Create the calculator instance
calculator = Calculator(root)

# Start the main loop of the application
root.mainloop()

```

Code Explanation

1. **Initialization:** The `Calculator` class is initialized with the main application window (`root`). It sets up the calculator display and buttons.
2. **Button creation:** Buttons are created in a 4x4 grid with text labels for digits, arithmetic operations, and clear (C) and equals (=) buttons.

3. **Button click handling:** When a button is clicked, the `on_button_click` method handles the event. It updates the display or performs the operation.
4. **Calculation:** The `calculate` method performs the arithmetic operation based on the given operands and operation.
5. **Main loop:** The application runs the main loop to handle events and updates.