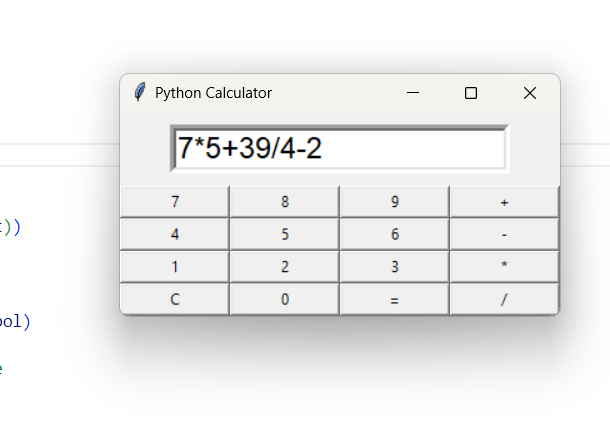
1. **Importing Tkinter**: The code starts by importing the Tkinter module as **tk**.
2. **Button Click Function**: The **click** function is defined to handle button clicks. It takes a **symbol** argument, which represents the button that was clicked. The function updates the display based on the button that was clicked.
3. **Creating the Main Window**: A new Tkinter window is created using **tk.Tk()**, and its title is set to "Python Calculator".
4. **Creating the Display Area**: A **StringVar** named **display\_var** is created to hold the display value. It is initialized to an empty string. An **Entry** widget named **display** is created to display the current value of **display\_var**. The **Entry** widget is configured with a font size of 18, a border width of 5, and it spans across all four columns of the window. It is also configured to expand and fill the available space in the window.
5. **Creating the Buttons**: Sixteen **Button** widgets are created to represent the digits 0-9, the arithmetic operators +, -, \*, and /, the equals sign =, and the clear button C. Each button is configured with a width of 5 and a command that calls the **click** function with the appropriate symbol when the button is clicked. The buttons are arranged in a 4x4 grid using the **grid** method, with each button occupying one cell in the grid.
6. **Running the Main Event Loop**: The **mainloop** method is called on the main window to start the Tkinter event loop, which listens for user input and updates the display accordingly.
7. The **try** and **except** statements in Python are used for error handling. The **try** block contains the code that may raise an exception, and the **except** block handles the exception if it occurs. This allows the program to gracefully handle errors and prevent the program from crashing.  
   the **eval** function is used to evaluate the mathematical expression entered by the user. If an error occurs during evaluation, such as a syntax error or division by zero, the **except** block is executed, and the display is set to 'Error'. This prevents the program from crashing and provides a user-friendly error message.

In summary, the code creates a simple calculator GUI with a display area and buttons for the digits 0-9, the arithmetic operators +, -, \*, and /, the equals sign =, and the clear button C. The **click** function is used to handle button clicks and update the display accordingly.

Conclusion - the Python Calculator project has provided us with a hands-on experience in creating a graphical user interface (GUI) application using the Tkinter library in Python. Through this project, we have learned the fundamentals of GUI programming, including creating widgets, handling events, and updating the display.



A screenshot of a calculator

Description automatically generated

# My Code –

import tkinter as tk

# buttons are clicked

def click(symbol):

    current = display\_var.get()

    if symbol == 'C':

        display\_var.set('')

    elif symbol == '=':

        try:

            result = eval(current)

            display\_var.set(str(result))

        except:

            display\_var.set('Error')

    else:

        display\_var.set(current + symbol)

# Defining variables and putting title

m = tk.Tk()

m.title("Python Calculator")

# Making Textbox to get input

display\_var = tk.StringVar()

display\_var.set('')

display = tk.Entry(m, textvariable=display\_var, font=('Lucida', 18), border=5)

display.grid(row=0, column=0, columnspan=4, sticky='news', padx=40, pady=10)

# Making buttons for calculator

button1 = tk.Button(m, text="7", width=5, command=lambda: click('7'))

button1.grid(row=1, column=0, sticky='news')

button2 = tk.Button(m, text="8", width=5, command=lambda: click('8'))

button2.grid(row=1, column=1, sticky='news')

button3 = tk.Button(m, text="9", width=5, command=lambda: click('9'))

button3.grid(row=1, column=2, sticky='news')

button4 = tk.Button(m, text="+", width=5, command=lambda: click('+'))

button4.grid(row=1, column=3, sticky='news')

# Next row

button5 = tk.Button(m, text="4", width=5, command=lambda: click('4'))

button5.grid(row=2, column=0, sticky='news')

button6 = tk.Button(m, text="5", width=5, command=lambda: click('5'))

button6.grid(row=2, column=1, sticky='news')

button7 = tk.Button(m, text="6", width=5, command=lambda: click('6'))

button7.grid(row=2, column=2, sticky='news')

button8 = tk.Button(m, text="-", width=5, command=lambda: click('-'))

button8.grid(row=2, column=3, sticky='news')

# Next row

button9 = tk.Button(m, text="1", width=5, command=lambda: click('1'))

button9.grid(row=3, column=0, sticky='news')

button10 = tk.Button(m, text="2", width=5, command=lambda: click('2'))

button10.grid(row=3, column=1, sticky='news')

button11 = tk.Button(m, text="3", width=5, command=lambda: click('3'))

button11.grid(row=3, column=2, sticky='news')

button12 = tk.Button(m, text="\*", width=5, command=lambda: click('\*'))

button12.grid(row=3, column=3, sticky='news')

# Next row

button13 = tk.Button(m, text="C", width=5, command=lambda: click('C'))

button13.grid(row=4, column=0, sticky='news')

button14 = tk.Button(m, text="0", width=5, command=lambda: click('0'))

button14.grid(row=4, column=1, sticky='news')

button15 = tk.Button(m, text="=", width=5, command=lambda: click('='))

button15.grid(row=4, column=2, sticky='news')

button16 = tk.Button(m, text="/", width=5, command=lambda: click('/'))

button16.grid(row=4, column=3, sticky='news')

# Run the main event loop

m.mainloop()