Final project- so far

This Python program creates a graphical user interface (GUI) using tkinter for solving algebraic equations with a single variable (x). It leverages the sympy library to parse, manipulate, and solve equations, displaying the solution and solving steps in a separate window. For now, I have been able to execute a version of the calculator that is able to solve equations but that’s the problem. This current version can *only* solve equations. I will try what I can and update it as best I can.

**Overlay**

GUI: Simple and intuitive interface with an entry field, a button, and a solution window.

Symbolic Solving: Uses sympy to handle linear and non-linear equations (e.g., quadratic equations like x\*\*2 = 4).

Step-by-Step Explanation: Shows the equation, simplification, and solutions in a clear format.

Error Handling: Gracefully handles invalid inputs with user-friendly error messages.

Extensibility: Can be extended to support other variables or more complex equations (though currently limited to x).

**Limitations**

Single Variable: Only solves for x. To support other variables (e.g., y), you’d need to modify the symbols definition and parsing logic.

Input Format: Requires Python-compatible syntax (e.g., x\*\*2 instead of x^2, \* for multiplication).

Simplistic Steps: The step-by-step explanation assumes moving all terms to one side. More complex equations (e.g., trigonometric or logarithmic) may need more detailed steps.

No Input Validation: Doesn’t check for malformed equations beyond what sympy parsing handles.

**Code**

import tkinter as tk

from tkinter import messagebox

from sympy import symbols, Eq, solve, simplify, pretty

from sympy.parsing.sympy\_parser import parse\_expr

x = symbols('x') # We'll assume the equations are in terms of x for simplicity

class CalculatorApp:

def \_\_init\_\_(self, root):

self.root = root

self.root.title("Equation Solver")

self.root.geometry("400x200")

self.label = tk.Label(root, text="Enter an equation (e.g., 2\*x + 3 = 7):")

self.label.pack(pady=10)

self.entry = tk.Entry(root, width=40)

self.entry.pack(pady=5)

self.solve\_button = tk.Button(root, text="Solve", command=self.solve\_equation)

self.solve\_button.pack(pady=10)

def solve\_equation(self):

user\_input = self.entry.get()

try:

left\_side, right\_side = user\_input.split('=')

left\_expr = parse\_expr(left\_side)

right\_expr = parse\_expr(right\_side)

equation = Eq(left\_expr, right\_expr)

solution = solve(equation, x)

steps = f"Solving the equation:\n\n"

steps += f"{pretty(equation)}\n\n"

steps += f"Subtract right side from both sides:\n"

combined = simplify(left\_expr - right\_expr)

steps += f"{pretty(combined)} = 0\n\n"

steps += "Solving for x:\n"

for sol in solution:

steps += f"x = {sol}\n"

self.show\_solution\_window(steps)

except Exception as e:

messagebox.showerror("Error", f"Invalid equation.\nDetails: {str(e)}")

def show\_solution\_window(self, steps):

solution\_win = tk.Toplevel(self.root)

solution\_win.title("Solution Steps")

solution\_win.geometry("500x300")

text\_box = tk.Text(solution\_win, wrap=tk.WORD, font=("Courier", 12))

text\_box.insert(tk.END, steps)

text\_box.pack(expand=True, fill=tk.BOTH)

if \_\_name\_\_ == "\_\_main\_\_":

root = tk.Tk()

app = CalculatorApp(root)

root.mainloop()