import tkinter as tk

import tkinter.messagebox as messagebox

import time

import math

import threading

MAX\_ITERATIONS = 200 # Set the maximum number of iterations

class CPUBenchmarkApp:

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

self.root = root

self.root.title("CPU Benchmark Tool")

self.root.geometry("500x350+600+250") # Set the window size

self.root.configure(bg="#F4EEEE") # Set background color

self.iterations\_var = tk.StringVar(value="20") # Use a string variable

self.score\_var = tk.StringVar(value="")

self.time\_var = tk.StringVar(value="")

self.create\_widgets()

def create\_widgets(self):

header\_label = tk.Label(self.root, text="CPU Benchmark Tool", font=(

"Arial", 20, "bold"), bg="#F4EEEE", fg="#2D2D2D")

header\_label.pack(pady=15)

iterations\_frame = tk.Frame(self.root, bg="#F4EEEE")

iterations\_frame.pack()

iterations\_label = tk.Label(iterations\_frame, text="Iterations(20-200):", font=(

"Arial", 14), bg="#F4EEEE", fg="#2D2D2D")

iterations\_label.pack(side="left", padx=10)

iterations\_entry = tk.Entry(

iterations\_frame, textvariable=self.iterations\_var, font=("Arial", 14), width=5)

iterations\_entry.pack(side="left")

# Add a validation function to limit the number of iterations

vcmd = (iterations\_entry.register(self.validate\_iterations), '%P')

iterations\_entry.config(validate="key", validatecommand=vcmd)

run\_button = tk.Button(self.root, text="Run Benchmark", command=self.run\_benchmark, font=(

"Arial", 14), bg="#2D2D2D", fg="white")

run\_button.pack(pady=20)

result\_frame = tk.Frame(self.root, bg="#F4EEEE")

result\_frame.pack()

score\_label = tk.Label(result\_frame, text="Score:", font=(

"Arial", 16, "bold"), bg="#F4EEEE", fg="#2D2D2D")

score\_label.pack()

self.score\_label = tk.Label(result\_frame, textvariable=self.score\_var, font=(

"Arial", 16), bg="#F4EEEE", fg="#2D2D2D")

self.score\_label.pack()

time\_label = tk.Label(result\_frame, text="Time (ms):", font=(

"Arial", 16, "bold"), bg="#F4EEEE", fg="#2D2D2D")

time\_label.pack()

self.time\_label = tk.Label(result\_frame, textvariable=self.time\_var, font=(

"Arial", 16), bg="#F4EEEE", fg="#2D2D2D")

self.time\_label.pack()

def validate\_iterations(self, new\_value):

if new\_value == "" or (new\_value.isnumeric() and 1 <= int(new\_value) <= MAX\_ITERATIONS):

return True

else:

return False

def run\_benchmark(self):

iterations = self.iterations\_var.get()

# Check if the input is empty or not a valid number

if not iterations or not iterations.isnumeric() or int(iterations) < 20:

messagebox.showerror("Error", "Please enter a valid number of iterations (20-200).")

return

self.score\_var.set("Running benchmark...")

self.time\_var.set("") # Clear the time taken value

def benchmark\_task():

start\_time = time.time()

for \_ in range(int(iterations)):

result = 0

for i in range(1, 1000000):

result += math.sqrt(i)

end\_time = time.time()

elapsed\_time = end\_time - start\_time

elapsed\_time\_ms = elapsed\_time \* 1000 # Convert to milliseconds

# Scoring mechanism: Lower time is better

score = 100 / elapsed\_time

self.score\_var.set(f"Score: {score:.2f} out of 100")

self.time\_var.set(f"{elapsed\_time\_ms:.2f} ms")

thread = threading.Thread(target=benchmark\_task)

thread.start()

def main():

root = tk.Tk()

CPUBenchmarkApp(root)

root.mainloop()

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

main()