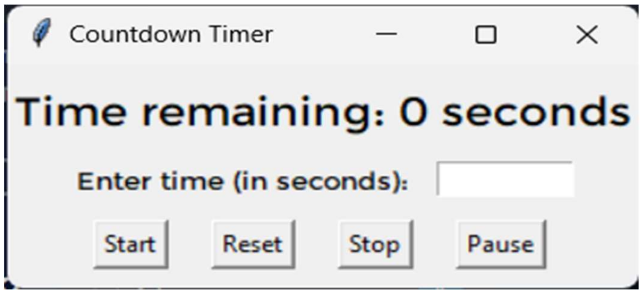
<u>Program</u> :-

```python
import tkinter as tk
from tkinter import messagebox
import threading

class CountdownTimer:
    def __init__(self, master):
        self.master = master
        self.master.title("Countdown Timer")
        self.time_remaining = 0
        self.is_paused = False

        # Create widgets
        self.label_time = tk.Label(self.master, text="Time remaining: 0
seconds", font=("Montserrat", 16))
        self.label_time.pack(pady=10)

        self.frame_input = tk.Frame(self.master)
        self.frame_input.pack()

        self.label_input = tk.Label(self.frame_input, text="Enter time (in
seconds):",font=("Montserrat", 10))
        self.label_input.pack(side=tk.LEFT, padx=5)

        self.entry_time = tk.Entry(self.frame_input, width=10)
        self.entry_time.pack(side=tk.LEFT, padx=5)

        self.button_start = tk.Button(self.master, text="Start",
command=self.start_timer)
        self.button_start.pack(side=tk.LEFT,pady=10,padx=(40,10))

        self.button_reset = tk.Button(self.master, text="Reset",
command=self.reset_timer)
        self.button_reset.pack(side=tk.LEFT,pady=5,padx=10)

        self.button_stop = tk.Button(self.master, text="Stop",
command=self.stop_timer)
        self.button_stop.pack(side=tk.LEFT,pady=5,padx=10)

        self.button_pause_resume = tk.Button(self.master, text="Pause",
command=self.pause_resume_timer)
        self.button_pause_resume.pack(side=tk.LEFT,pady=5,padx=10)

    def start_timer(self):
        if self.time_remaining > 0:
            messagebox.showwarning("Warning", "Timer is already running!")
            return
        try:
            self.time_remaining = int(self.entry_time.get())
        except ValueError:
            messagebox.showerror("Error", "Invalid input. Please enter an
integer.")
            return

        if self.time_remaining <= 0:
            messagebox.showerror("Error", "Invalid input. Please enter a
positive integer.")
            return

        self.update_label()

        # Start a new thread to run the countdown
        self.thread_timer = threading.Thread(target=self.run_timer)
        self.thread_timer.start()

    def reset_timer(self):
        self.time_remaining = 0
        self.is_paused = False
        self.update_label()

    def stop_timer(self):
        self.time_remaining = 0
        self.is_paused = False
        self.update_label()
        messagebox.showinfo("Info", "Timer stopped!")

    def pause_resume_timer(self):
        if self.time_remaining <= 0:
            messagebox.showwarning("Warning", "Timer is not running!")
            return

        if self.is_paused:
            self.is_paused = False
            self.button_pause_resume.config(text="Pause")
        else:
            self.is_paused = True
            self.button_pause_resume.config(text="Resume")

    def run_timer(self):
        while self.time_remaining > 0:
            if not self.is_paused:
                self.time_remaining -= 1
                self.update_label()
            threading.Event().wait(1)
        messagebox.showinfo("Info", "Time's up!")
        self.reset_timer()

    def update_label(self):
        self.label_time.config(text="Time remaining: {}
seconds".format(self.time_remaining))

if __name__ == "__main__":
    root = tk.Tk()
    app = CountdownTimer(root)
    root.mainloop()
```

<u>Output</u> :-



Time remaining: 0 seconds

Enter time (in seconds):

A Countdown Timer created with features such as Reset/Stop and Pause/Resume.

Github :-
https://github.com/rovxn/countdown_timer_python

Submitted by

Rovan Thomas John

Batch :- March 2023