Python Minor Project

Submitted by: Alavala Sai Goutham Reddy.

Email Id: gouthamsai.alavala@gmail.com

Problem Statement:

Create A Countdown Timer Using Python including features: Reset/Stop, Pause/Resume.

Solution:

The python code for creating a countdown timer which includes the features as Reset/Stop, Pause/Resume is as followed:

```
import tkinter as tk
import threading
import time

class CountdownTimer:
    def __init__(x, minutes, main):
        x.main = main
        x.minutes = minutes
        x.seconds = minutes * 60
        x.is_running = False
        x.is_paused = False

        x.label = tk.Label(x.main, text=x.format_time(), font=('Futura', 68))
        x.label.pack()
```

```
        button_frame = tk.Frame(x.main)
        button_frame.pack(padx=100, pady=100)

        x.start_button = tk.Button(button_frame, text='Start', command=x.start,
font=('Futura', 20))
        x.start_button.pack(side=tk.LEFT, padx=6)

        x.reset_button = tk.Button(button_frame, text='Reset', command=x.reset,
state=tk.DISABLED, font=('Futura', 20))
        x.reset_button.pack(side=tk.LEFT, padx=6)

        x.pause_button = tk.Button(button_frame, text='Pause', command=x.pause,
state=tk.DISABLED, font=('Futura', 20))
        x.pause_button.pack(side=tk.LEFT, padx=6)

        x.resume_button = tk.Button(button_frame, text='Resume', command=x.resume,
state=tk.DISABLED, font=('Futura', 20))
        x.resume_button.pack(side=tk.LEFT, padx=6)

        x.timer_thread = None

    def start(x):
        if x.is_running:
            return
        x.is_running = True
        x.start_button.config(state=tk.DISABLED)
        x.reset_button.config(state=tk.NORMAL)
```

```
    x.pause_button.config(state=tk.NORMAL)
    x.timer_thread = threading.Thread(target=x.run_timer)
    x.timer_thread.start()

def reset(x):
    x.is_running = False
    x.is_paused = False
    x.seconds = x.minutes * 60
    x.label.config(text=x.format_time())
    x.start_button.config(state=tk.NORMAL)
    x.reset_button.config(state=tk.DISABLED)
    x.pause_button.config(state=tk.DISABLED)
    x.resume_button.config(state=tk.DISABLED)

def pause(x):
    x.is_paused = True
    x.pause_button.config(state=tk.DISABLED)
    x.resume_button.config(state=tk.NORMAL)

def resume(x):
    x.is_paused = False
    x.resume_button.config(state=tk.DISABLED)
    x.pause_button.config(state=tk.NORMAL)

def run_timer(x):
    while x.is_running and x.seconds > 0:
        if not x.is_paused:
```

```
            x.seconds -= 1
            x.label.config(text=x.format_time())
        time.sleep(1)
    if x.is_running:
        x.is_running = False
        x.label.config(text="Time's up!")
        x.start_button.config(state=tk.NORMAL)
        x.reset_button.config(state=tk.DISABLED)
        x.pause_button.config(state=tk.DISABLED)
        x.resume_button.config(state=tk.DISABLED)

  def format_time(x):
      minutes, seconds = divmod(x.seconds, 60)
      return f"{minutes:02d}:{seconds:02d}"

# Example usage:
main = tk.Tk()
main.title('Hello. Thank for using countdown timer')

entry_frame = tk.Frame(main)
entry_frame.pack(padx=100, pady=100)

label = tk.Label(entry_frame, text='Please enter number of minutes for countdown:',
font=('Garamond', 20))
label.pack(side=tk.LEFT)

entry = tk.Entry(entry_frame)
```

entry.pack(side=tk.LEFT)

start_button = tk.Button(main, text='Countdown', command=lambda: CountdownTimer(int(entry.get()), main), font=('Garamond', 20))

start_button.pack()

main.mainloop()

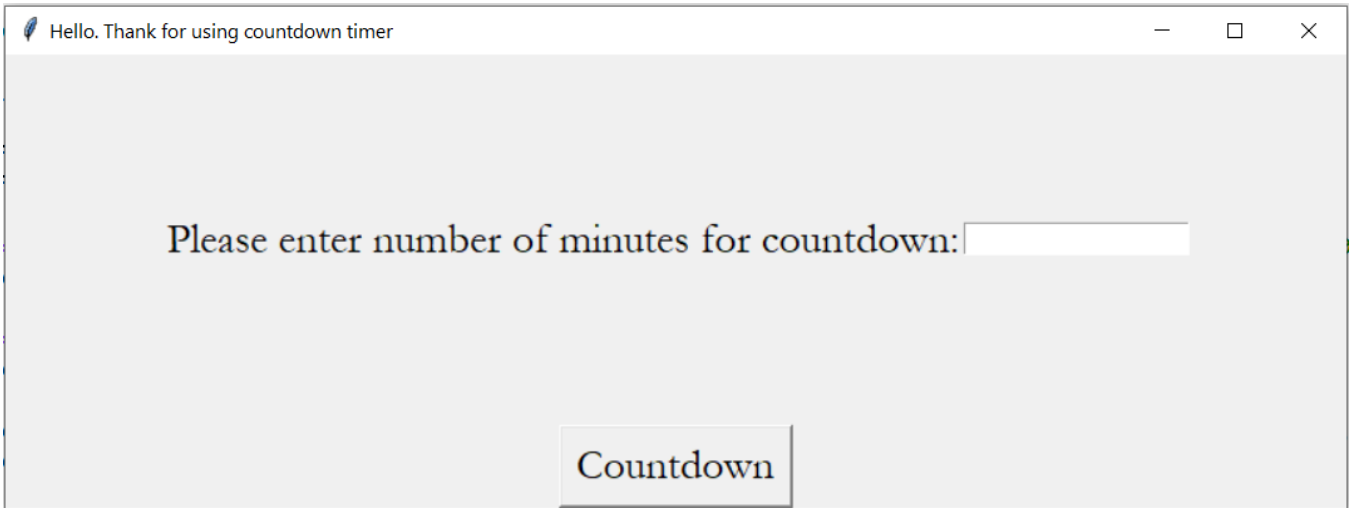
```
import tkinter as tk
import threading
import time

class CountdownTimer:
    def __init__(x, minutes, main):
        x.main = main
        x.minutes = minutes
        x.seconds = minutes * 60
        x.is_running = False
        x.is_paused = False

        x.label = tk.Label(x.main, text=x.format_time(), font=('Futura', 68))
        x.label.pack()

        button_frame = tk.Frame(x.main)
        button_frame.pack(padx=100, pady=100)

        x.start_button = tk.Button(button_frame, text='Start', command=x.start, font=('Futura', 20))
        x.start_button.pack(side=tk.LEFT, padx=6)

        x.reset_button = tk.Button(button_frame, text='Reset', command=x.reset, state=tk.DISABLED, font=('Futura', 20))
        x.reset_button.pack(side=tk.LEFT, padx=6)

        x.pause_button = tk.Button(button_frame, text='Pause', command=x.pause, state=tk.DISABLED, font=('Futura', 20))
        x.pause_button.pack(side=tk.LEFT, padx=6)

        x.resume_button = tk.Button(button_frame, text='Resume', command=x.resume, state=tk.DISABLED, font=('Futura', 20))
        x.resume_button.pack(side=tk.LEFT, padx=6)

        x.timer_thread = None
```

```
        x.resume_button = tk.Button(button_frame, text='Resume', command=x.resume, state=tk.DISABLED, font=('Futura', 20))
        x.resume_button.pack(side=tk.LEFT, padx=6)

        x.timer_thread = None

    def start(x):
        if x.is_running:
            return
        x.is_running = True
        x.start_button.config(state=tk.DISABLED)
        x.reset_button.config(state=tk.NORMAL)
        x.pause_button.config(state=tk.NORMAL)
        x.timer_thread = threading.Thread(target=x.run_timer)
        x.timer_thread.start()

    def reset(x):
        x.is_running = False
        x.is_paused = False
        x.seconds = x.minutes * 60
        x.label.config(text=x.format_time())
        x.start_button.config(state=tk.NORMAL)
        x.reset_button.config(state=tk.DISABLED)
        x.pause_button.config(state=tk.DISABLED)
        x.resume_button.config(state=tk.DISABLED)

    def pause(x):
        x.is_paused = True
        x.pause_button.config(state=tk.DISABLED)
        x.resume_button.config(state=tk.NORMAL)

    def resume(x):
```

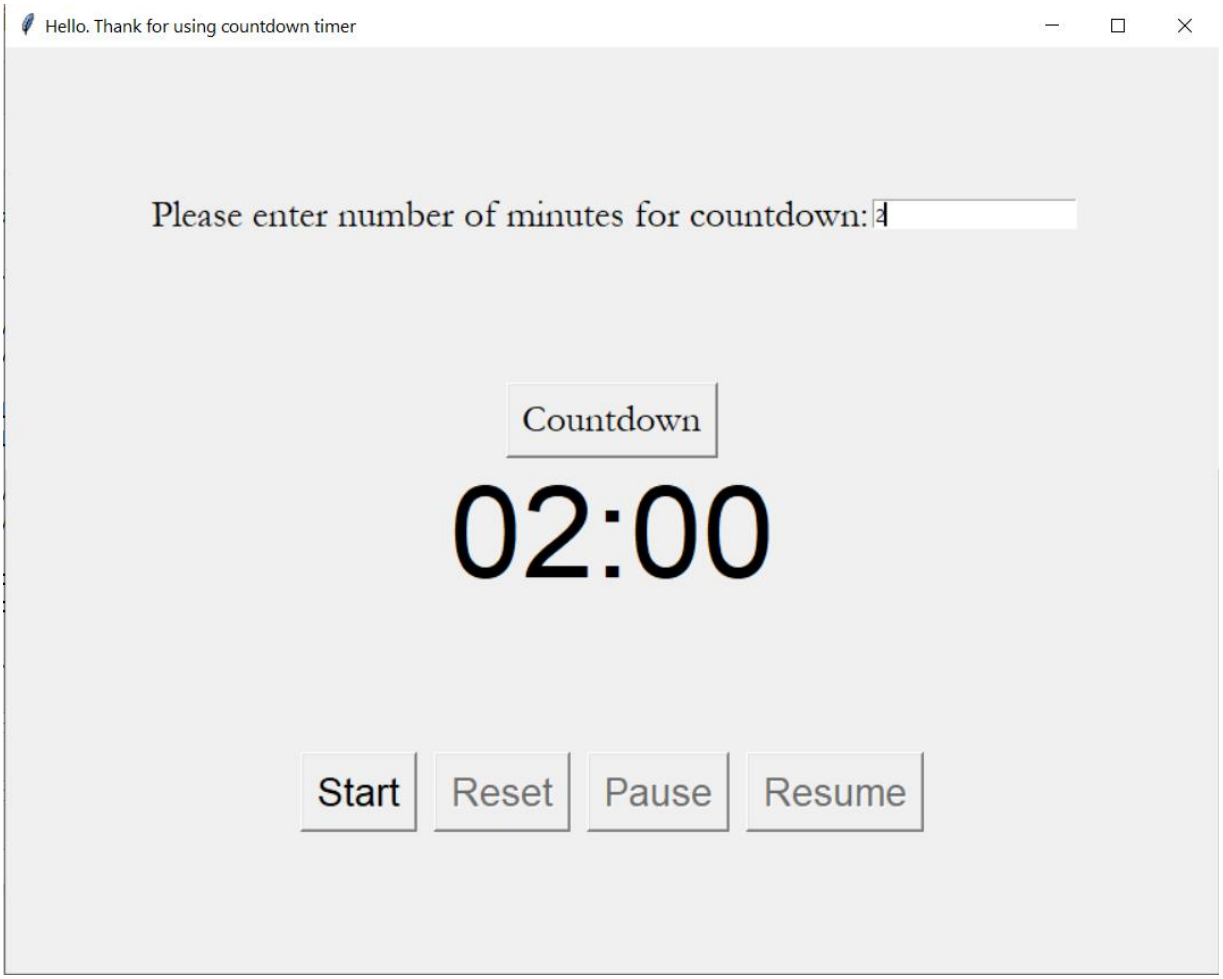
```
    def resume(x):
        x.is_paused = False
        x.resume_button.config(state=tk.DISABLED)
        x.pause_button.config(state=tk.NORMAL)

    def run_timer(x):
        while x.is_running and x.seconds > 0:
            if not x.is_paused:
                x.seconds -= 1
                x.label.config(text=x.format_time())
            time.sleep(1)
        if x.is_running:
            x.is_running = False
            x.label.config(text="Time's up!")
            x.start_button.config(state=tk.NORMAL)
            x.reset_button.config(state=tk.DISABLED)
            x.pause_button.config(state=tk.DISABLED)
            x.resume_button.config(state=tk.DISABLED)

    def format_time(x):
        minutes, seconds = divmod(x.seconds, 60)
        return f"{minutes:02d}:{seconds:02d}"

# Example usage:
main = tk.Tk()
main.title('Hello. Thank for using countdown timer')

entry_frame = tk.Frame(main)
entry_frame.pack(padx=100, pady=100)

label = tk.Label(entry_frame, text='Please enter number of minutes for countdown:', font=('Garamond', 20))
```

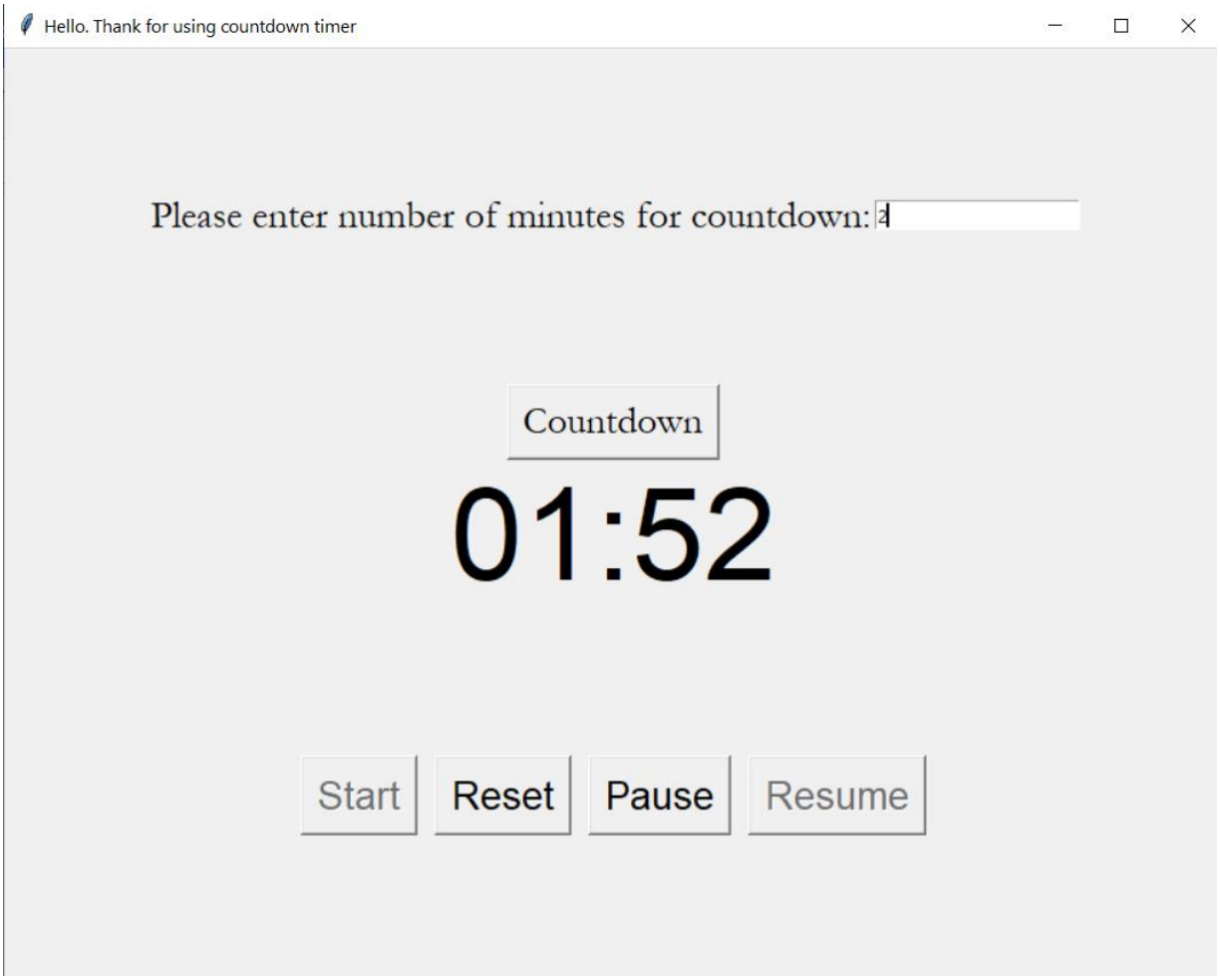
```
            time.sleep(1)
        if x.is_running:
            x.is_running = False
            x.label.config(text="Time's up!")
            x.start_button.config(state=tk.NORMAL)
            x.reset_button.config(state=tk.DISABLED)
            x.pause_button.config(state=tk.DISABLED)
            x.resume_button.config(state=tk.DISABLED)

    def format_time(x):
        minutes, seconds = divmod(x.seconds, 60)
        return f"{minutes:02d}:{seconds:02d}"

main = tk.Tk()
main.title('Hello. Thank for using countdown timer')

entry_frame = tk.Frame(main)
entry_frame.pack(padx=100, pady=100)

label = tk.Label(entry_frame, text='Please enter number of minutes for countdown:', font=('Garamond', 20))
label.pack(side=tk.LEFT)

entry = tk.Entry(entry_frame)
entry.pack(side=tk.LEFT)

start_button = tk.Button(main, text='Countdown', command=lambda: CountdownTimer(int(entry.get()), main), font=('Garamond', 20))
start_button.pack()

main.mainloop()
```

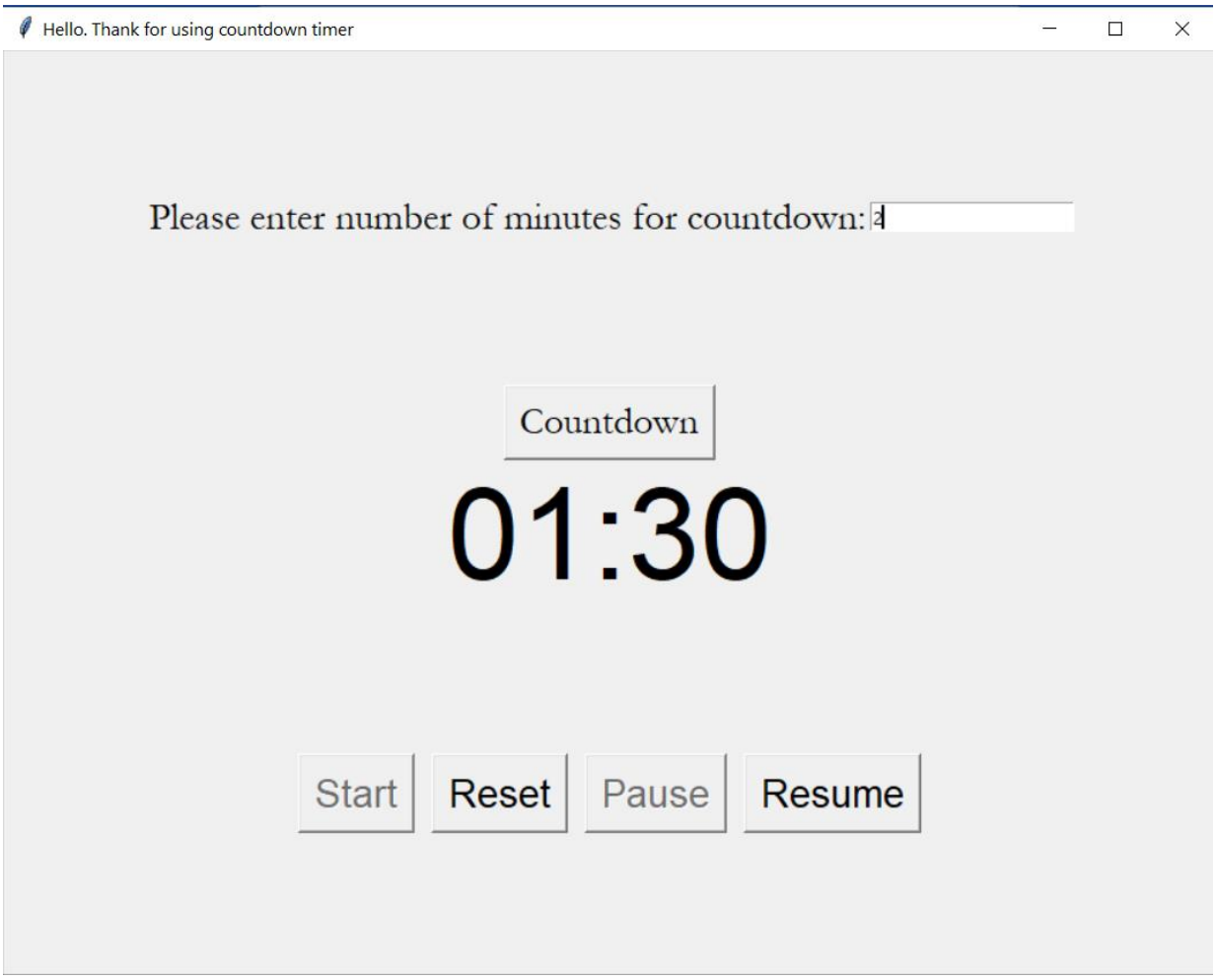
By running the following code in the jupyter notebook platform, we will be obtaining a pop-up window, which allows user to enter the minutes for the countdown to be set.



After entering the desired countdown timer, by clicking on the "**Countdown**" option on the window, the extended window appears with all the required features.

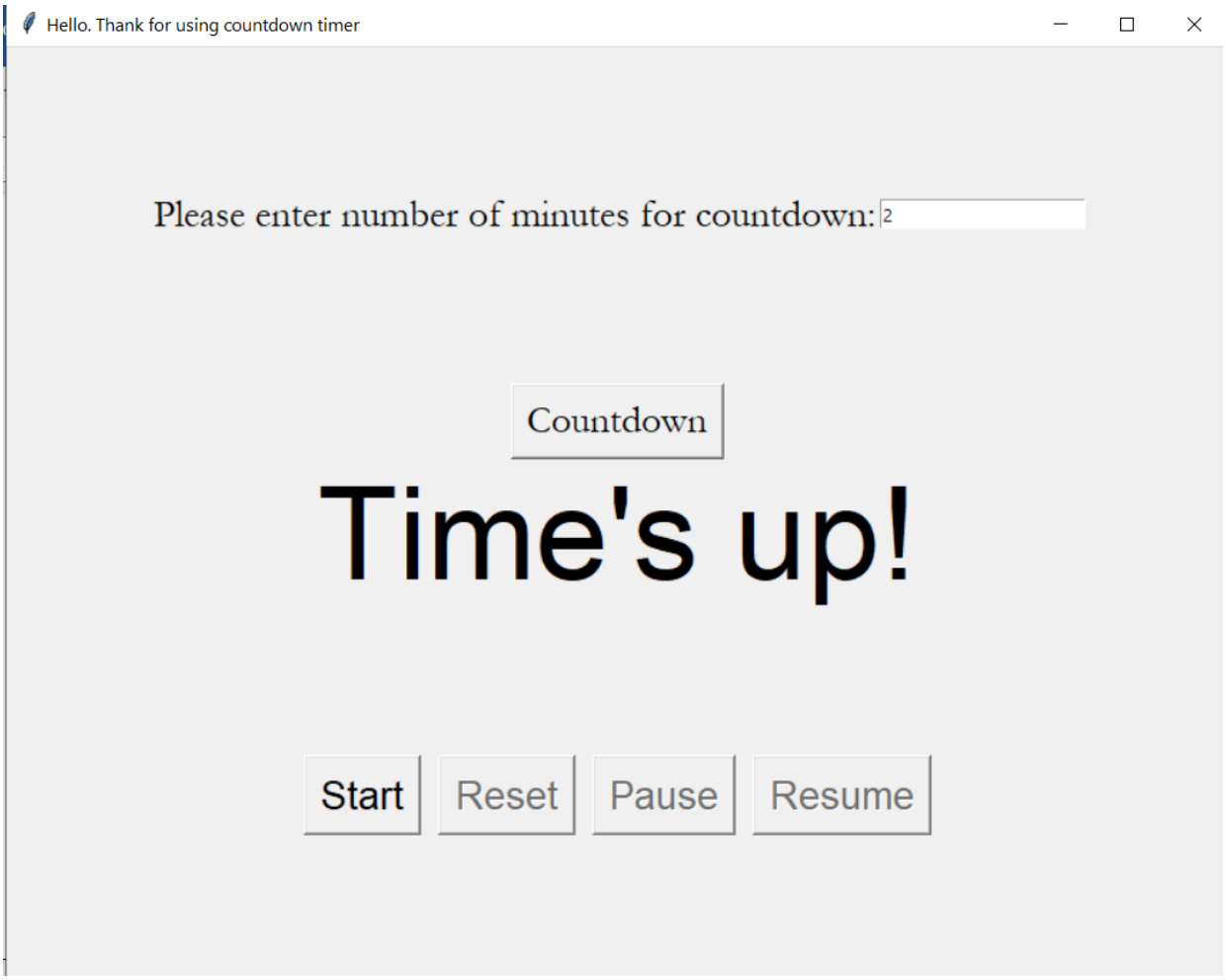

Hello. Thank for using countdown timer
Please enter number of minutes for countdown: 2
Countdown
02:00
Start
Reset
Pause
Resume


Hello. Thank for using countdown timer
Please enter number of minutes for countdown: 2
Countdown
01:52
Start
Reset
Pause
Resume



The features are enabled at any time of point.

And when the time runs out, it displays "**Time's up!**".



In this way we can create a countdown timer with the features we need.

Link for jupyter notebook: Drive link for codes.