Aquino,Aaron Jan O.
C204

Finals Lab Task 4. Python GUI using TKINTER

Note: Write your code following OOP code construct, you may use the attached simpleCalc.py program as guide.

3. Make a program that will Allow the user to Select Destination Code (between 1 – 4) using ComboBox widget, A Time Code using radio buttons, And the Duration Of The Call in minutes and output the TOTAL CHARGE. – Validate user inputs by using TRY EXCEPT block – Only numeric values are accepted.

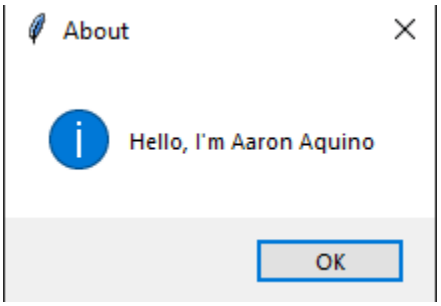
4. Compute Button should compute for the TOTAL CHARGE.

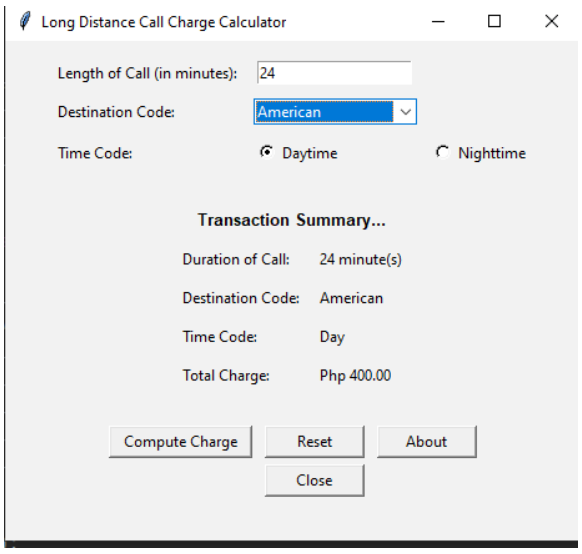
4.1 Computations should be based on the table rates shown above. (The total charge is based on Length of Calls, Destination Code and Time Code)

4.2. You may use the get () method of the comboBox to capture the selected option in your comboBox

5. Reset Button should clear the Radio Button Selection and the Text field entries should be cleared as well

6. About button should display a dialog with the message: “Hello I’m your Name”





```python
import tkinter as tk
from tkinter import ttk, messagebox


DAY_RATES = {
    1: (50, 3),  # American: P50 every 3 minutes
    2: (30, 2),  # Asian: P30 every 2 minutes
    3: (40, 3),  # African: P40 every 3 minutes
    4: (35, 2)   # European: P35 every 2 minutes
}

NIGHT_RATES = {
    1: (45, 3),  # American: P45 every 3 minutes
    2: (27, 2),  # Asian: P27 every 2 minutes
    3: (36, 3),  # African: P36 every 3 minutes
    4: (30, 2)   # European: P30 every 2 minutes
}


1 usage
class CallChargeCalculator:
    def __init__(self, minutes, dest_code, time_code):
        self.minutes = minutes
        self.dest_code = dest_code
        self.time_code = time_code

    1 usage
    def compute_charge(self):
        if self.time_code == "Day":
            rate, per_minutes = DAY_RATES[self.dest_code]
        else:
            rate, per_minutes = NIGHT_RATES[self.dest_code]


        intervals = (self.minutes + per_minutes - 1) // per_minutes
        total = intervals * rate
        return total
```

```
class CallChargeCalculatorApp:
    def __init__(self, root):
        self.root = root
        self.root.title("Long Distance Call Charge Calculator")
        self.root.geometry("380x400")
        self.root.config(bg="#f2f2f2")  # Light gray background


        self.create_widgets()

    1 usage
    def create_widgets(self):

        input_frame = tk.Frame(self.root, bg="#f2f2f2")
        input_frame.pack(pady=10)


        tk.Label(input_frame, text="Length of Call (in minutes):", bg="#f2f2f2").grid(row=0, column=0, padx=5, pady=5, sticky="w")
        self.entry_minutes = tk.Entry(input_frame, width=20)
        self.entry_minutes.grid(row=0, column=1, padx=5, pady=5)

        tk.Label(input_frame, text="Destination Code:", bg="#f2f2f2").grid(row=1, column=0, padx=5, pady=5, sticky="w")
        self.combo_dest = ttk.Combobox(input_frame, values=["American", "Asian", "African", "European"], state="readonly", width=18)
        self.combo_dest.grid(row=1, column=1, padx=5, pady=5)


        tk.Label(input_frame, text="Time Code:", bg="#f2f2f2").grid(row=2, column=0, padx=5, pady=5, sticky="w")

        self.time_var = tk.StringVar()
        tk.Radiobutton(input_frame, text="Daytime", variable=self.time_var, value="Day", bg="#f2f2f2").grid(row=2, column=1, padx=5, pady=5, sticky="w")
        tk.Radiobutton(input_frame, text="Nighttime", variable=self.time_var, value="Night", bg="#f2f2f2").grid(row=2, column=2, padx=5, pady=5, sticky="w")


        self.summary_frame = tk.Frame(self.root, bg="#f2f2f2")
        self.summary_frame.pack(pady=10)
```

```
        self.transaction_label = tk.Label(self.summary_frame, text="Transaction Summary...", bg="#f2f2f2", font=("Arial", 10, "bold"))
        self.transaction_label.grid(row=0, column=0, columnspan=2, padx=5, pady=5)

        self.label_duration = tk.Label(self.summary_frame, text="Duration of Call:", bg="#f2f2f2")
        self.label_duration.grid(row=1, column=0, padx=5, pady=5, sticky="w")
        self.label_duration_value = tk.Label(self.summary_frame, text="", bg="#f2f2f2")
        self.label_duration_value.grid(row=1, column=1, padx=5, pady=5, sticky="w")

        self.label_destination = tk.Label(self.summary_frame, text="Destination Code:", bg="#f2f2f2")
        self.label_destination.grid(row=2, column=0, padx=5, pady=5, sticky="w")
        self.label_destination_value = tk.Label(self.summary_frame, text="", bg="#f2f2f2")
        self.label_destination_value.grid(row=2, column=1, padx=5, pady=5, sticky="w")

        self.label_time_code = tk.Label(self.summary_frame, text="Time Code:", bg="#f2f2f2")
        self.label_time_code.grid(row=3, column=0, padx=5, pady=5, sticky="w")
        self.label_time_code_value = tk.Label(self.summary_frame, text="", bg="#f2f2f2")
        self.label_time_code_value.grid(row=3, column=1, padx=5, pady=5, sticky="w")

        self.label_total_charge = tk.Label(self.summary_frame, text="Total Charge:", bg="#f2f2f2")
        self.label_total_charge.grid(row=4, column=0, padx=5, pady=5, sticky="w")
        self.label_total_charge_value = tk.Label(self.summary_frame, text="", bg="#f2f2f2")
        self.label_total_charge_value.grid(row=4, column=1, padx=5, pady=5, sticky="w")


        btn_frame = tk.Frame(self.root, bg="#f2f2f2")
        btn_frame.pack(pady=15)

        tk.Button(btn_frame, text="Compute Charge", width=15, command=self.compute_charge).grid(row=0, column=0, padx=5)
        tk.Button(btn_frame, text="Reset", width=10, command=self.reset_all).grid(row=0, column=1, padx=5)
        tk.Button(btn_frame, text="About", width=10, command=self.about).grid(row=0, column=2, padx=5)
        tk.Button(btn_frame, text="Close", width=10, command=self.root.quit).grid(row=1, column=1, padx=5, pady=5)

    1 usage
```

```python
    def compute_charge(self):

        try:
            minutes = int(self.entry_minutes.get())
            if minutes <= 0:
                raise ValueError
        except ValueError:
            messagebox.showerror( title: "Error",  message: "Please enter a VALID numeric value for minutes.")
            return


        dest_label = self.combo_dest.get()
        destination_map = {"American": 1, "Asian": 2, "African": 3, "European": 4}
        dest_code = destination_map.get(dest_label)

        if not dest_code:
            messagebox.showerror( title: "Error",  message: "Please select a valid destination.")
            return


        time_code = self.time_var.get()
        if not time_code:
            messagebox.showerror( title: "Error",  message: "Please select Daytime or Nighttime.")
            return


        calculator = CallChargeCalculator(minutes, dest_code, time_code)
        total_charge = calculator.compute_charge()


        self.label_duration_value.config(text=f"{minutes} minute(s)")
        self.label_destination_value.config(text=dest_label)
        self.label_time_code_value.config(text=time_code)
        self.label_total_charge_value.config(text=f"Php {total_charge:.2f}")
```

```
    1 usage
    def reset_all(self):

        self.combo_dest.set("")
        self.time_var.set("")
        self.entry_minutes.delete( first: 0, tk.END)
        self.label_duration_value.config(text="")
        self.label_destination_value.config(text="")
        self.label_time_code_value.config(text="")
        self.label_total_charge_value.config(text="")

    1 usage
    def about(self):
        messagebox.showinfo( title: "About",  message: "Hello, I'm Aaron Aquino")


#RUN PROGRAM
if __name__ == "__main__":
    root = tk.Tk()
    app = CallChargeCalculatorApp(root)
    root.mainloop()
```