Let's go through the Python alarm clock code line by line:

Importing Required Libraries

```
import tkinter as Tk

import time

from tkinter import messagebox
```

- import tkinter as Tk: This imports the **Tkinter** library, which is used to create graphical user interfaces (GUIs). We alias it as Tk for convenience.
- import time: The **time** module allows us to work with the system clock, including getting the current time and adding delays.
- from tkinter import messagebox: **messagebox** is a part of Tkinter used to display pop-up messages (e.g., when the alarm goes off).

Defining the Alarm Function

```
def actual_time():

  set_alarm_time = f"{hourTime.get()}:{minTime.get()}:{secTime.get()}"
```

- def actual_time(): Defines a function called actual_time. This function will be executed when you click the "Set Alarm" button.
- set_alarm_time = f"{hourTime.get()}:{minTime.get()}:{secTime.get()}":
  - This creates a string set_alarm_time in the format HH:MM:SS (hour, minute, second).
  - hourTime.get(), minTime.get(), and secTime.get() fetch the user input from the entry boxes.

```
while True:

  time.sleep(1)

  current_time = time.strftime("%H:%M:%S")

  print(f"Current Time: {current_time}")
```

- while True: Creates an infinite loop that continuously checks the current time.
- time.sleep(1): Pauses the loop for 1 second. This prevents the program from using excessive CPU power and ensures the alarm time is checked every second.
- current_time = time.strftime("%H:%M:%S"): Retrieves the current system time in "HH:MM

" format.

- print(f"Current Time: {current_time}"): Prints the current time to the console (this is helpful for debugging or tracking when the alarm is about to go off).

```
  if current_time == set_alarm_time:

    messagebox.showinfo("Alarm", "Time to wake up!")

     break
```

- if current_time == set_alarm_time: Checks whether the current system time matches the alarm time entered by the user.
- messagebox.showinfo("Alarm", "Time to wake up!"): If the current time matches the alarm time, a pop-up message box appears with the message "Time to wake up!".
- break: Stops the loop once the alarm has gone off.

Tkinter Window Setup

clock = Tk.Tk()

clock.title("DataFlair Alarm Clock")

clock.geometry("400x200")

clock.configure(bg='#FFC0CB') # Setting a pink background color

- clock = Tk.Tk(): Initializes the main window (dialog box) of the alarm clock.
- clock.title("DataFlair Alarm Clock"): Sets the title of the window to "DataFlair Alarm Clock".
- clock.geometry("400x200"): Specifies the size of the window to be 400 pixels wide and 200 pixels tall.
- clock.configure(bg='#FFC0CB'): Changes the background color of the window to pink (#FFC0CB is the hex code for a light pink shade).

Adding Labels to the Window

time_format = Tk.Label(clock, text="Enter time in 24-hour format!", fg="white", bg="#FF69B4", font=("Arial", 15))

time_format.place(x=60, y=20)

- time_format = Tk.Label(...): Creates a label that displays the text "Enter time in 24-hour format!".
    - fg="white": Sets the text color to white.
    - bg="#FF69B4": Sets the background color of the label to a darker pink.
    - font=("Arial", 15): Sets the font style to Arial with a size of 15.
- time_format.place(x=60, y=20): Positions the label on the window at coordinates (60, 20).

---

addTime = Tk.Label(clock, text="Hour Min Sec", fg="white", bg="#FF69B4", font=("Arial", 12))

addTime.place(x=140, y=50)

- This label displays "Hour Min Sec" as a heading for the input boxes. It helps users understand where to enter the hour, minute, and second.

---

setYourAlarm = Tk.Label(clock, text="When to wake you up", fg="white", bg="#FF69B4", relief="solid", font=("Arial", 13, "bold"))

setYourAlarm.place(x=120, y=80)

- This label, "When to wake you up," is placed just above the input boxes. The relief="solid" adds a solid border around the label to make it stand out.

Creating Input Boxes

hourTime = Tk.StringVar()

minTime = Tk.StringVar()

secTime = Tk.StringVar()

- These variables (hourTime, minTime, secTime) store the user's input for the alarm time in hours, minutes, and seconds. Each is a **StringVar()**, meaning it can hold string data.

---

hourEntry = Tk.Entry(clock, textvariable=hourTime, bg="white", width=4, font=("Arial", 12))

hourEntry.place(x=140, y=120)

- Tk.Entry(...): Creates an input box where the user can enter the hour.
    - textvariable=hourTime: Links this input box to the hourTime variable.
    - bg="white": Sets the background color of the input box to white.
    - width=4: Limits the input box to 4 characters.
    - font=("Arial", 12): Sets the font style and size for the input.
- hourEntry.place(x=140, y=120): Positions the input box for hours at coordinates (140, 120).

---

minEntry = Tk.Entry(clock, textvariable=minTime, bg="white", width=4, font=("Arial", 12))

minEntry.place(x=190, y=120)

- Similar to the hour input box, this creates an input field for the minutes (minTime).

---

secEntry = Tk.Entry(clock, textvariable=secTime, bg="white", width=4, font=("Arial", 12))

secEntry.place(x=240, y=120)

- Similar to the other two, this input box is for entering the seconds (secTime).

Creating the Set Alarm Button

submit = Tk.Button(clock, text="Set Alarm", fg="white", bg="#FF69B4", width=10, command=actual_time, font=("Arial", 12))

submit.place(x=150, y=160)

- Tk.Button(...): Creates a button labeled "Set Alarm".
    - fg="white": Sets the text color to white.
    - bg="#FF69B4": Sets the background color to pink.
    - width=10: Specifies the width of the button.
    - command=actual_time: Links the button to the actual_time() function, so clicking it starts the alarm-checking process.

    - font=("Arial", 12): Sets the font style and size for the button text.
- submit.place(x=150, y=160): Positions the button at coordinates (150, 160).

Running the Tkinter Loop

clock.mainloop()

- clock.mainloop(): This command starts the Tkinter event loop, which listens for user interaction with the GUI (like clicking buttons or entering text). It keeps the window running until it's closed