

Class Assignment (CA) 1 Report

On

Alarm tool

Submitted in partial fulfillment of the requirements for the award of degree

In the department of

Computer Science and Engineering

LOVELY PROFESSIONAL UNIVERSITY PHAGWARA, PUNJAB



L OVELY
P ROFESSIONAL
U NIVERSITY

Transforming Education Transforming India

Submitted by:

- **Divya Jyothi. Chinnabathini. (Reg.No: 12007857)**
- **Sai Sandhya. Chennuru. (Reg.No: 12013893)**

Contents

S. No	Topic name	Page no
01	Details of student	01
02	contents	02
03	List of figures	03
04	abstract	04
05	Introduction	05
06	Details of project	06
07	Coding and screenshots of project	09
08	conclusion	11
09	references	13

List of figures

S.No	Figure name	Page no
1.	Alarm clock	12

Abstract

In this project, we are going to use some external modules which are already made available by other developers. These modules will help us save a lot of time and effort. All we have to do is import them into our project to get started.

Importing modules is pretty simple. All you have to do is run a simple pip install command from terminal & our specified module will be downloaded in our system.

We need 2 different modules for our project - date time & play sound.

Let's run pip install command and download both of these modules.
pip install
date time

Date time - We will use this module to obtain current time which is not possible without this module.

This project requires good knowledge of Python and GUI (Graphic User Interface). Python when combined with Tkinter provides a fast and easy way to create GUI applications. Tkinter provides a powerful object-oriented interface to the Tk GUI toolkit. All the modules used need not be downloaded beforehand like the other libraries like NumPy, thus this project will be user friendly and accessible in any virtual environment used for python programming.

Introduction

The objective of our project is to implement an alarm clock using Python. Python consists of some very innovative libraries such as date time and tkinter which help us to build the project using the current date and time as well as to provide a user interface to set the alarm according to the requirement in 24-hour format.

First, let's check the steps to build an Alarm Clock program in Python:

Importing all the libraries and modules required

Putting forward a while loop which takes the argument of the time, the user wants to set the alarm on and automatically breaks when the time is up, with sound to get input. With this project in Python, we have successfully made the Alarm Clock. We used the popular GUI library for rendering graphics on a display window. We learned how to extract the current time from the computer and to use it for manipulation using the Date Time library. This way we can set an alarm in the computer interface using python programming which rings with the default machine sound for Windows. Tkinter module belongs to a standard library of GUI in Python. It helps us to create a dialog box with any information that we want to provide or get from the users

Details of project

step1: Import Required Library

```
# Import Required Library
```

```
from tkinter import *
```

```
import datetime
```

```
import time
```

```
import winsound
```

Step 2: Add Button, Labels, Frame, and option menus.

```
# Button
```

```
Button (Object Name, text="Enter Text", **attr)
```

```
# Label
```

```
Label (Object Name, text="Enter Text", command="Enter Command",  
**attr)
```

```
# Frame
```

```
Frame (Object Name, **attr)
```

```
# Option Menu
```

```
Option Menu("Object Name", "Data Type", "list of value in form of  
tuple", **attr)
```

Step 3: Make a function named alarm (), which performs alarm clock work

```
def alarm ():
```

```
# Infinite Loop
```

```
while True:
```

```
    # Set Alarm
```

```
    set alarm = f"{hour. get():{minute. get():{second. get()}"
```

```
# Wait for one seconds
```

```
time. sleep (1)
```

```
# Get current time
```

```
current time = datetime.datetime.now ().strftime("%H:%M:%S")
```

```
# Check whether set alarm is equal to current time or not
```

```
if current time == set alarm:
```

```
    print("Time to Wake up")
```

```
# Playing sound
```

```
winsound.PlaySound ("sound.wav",winsound.SND_ASYNC)
```

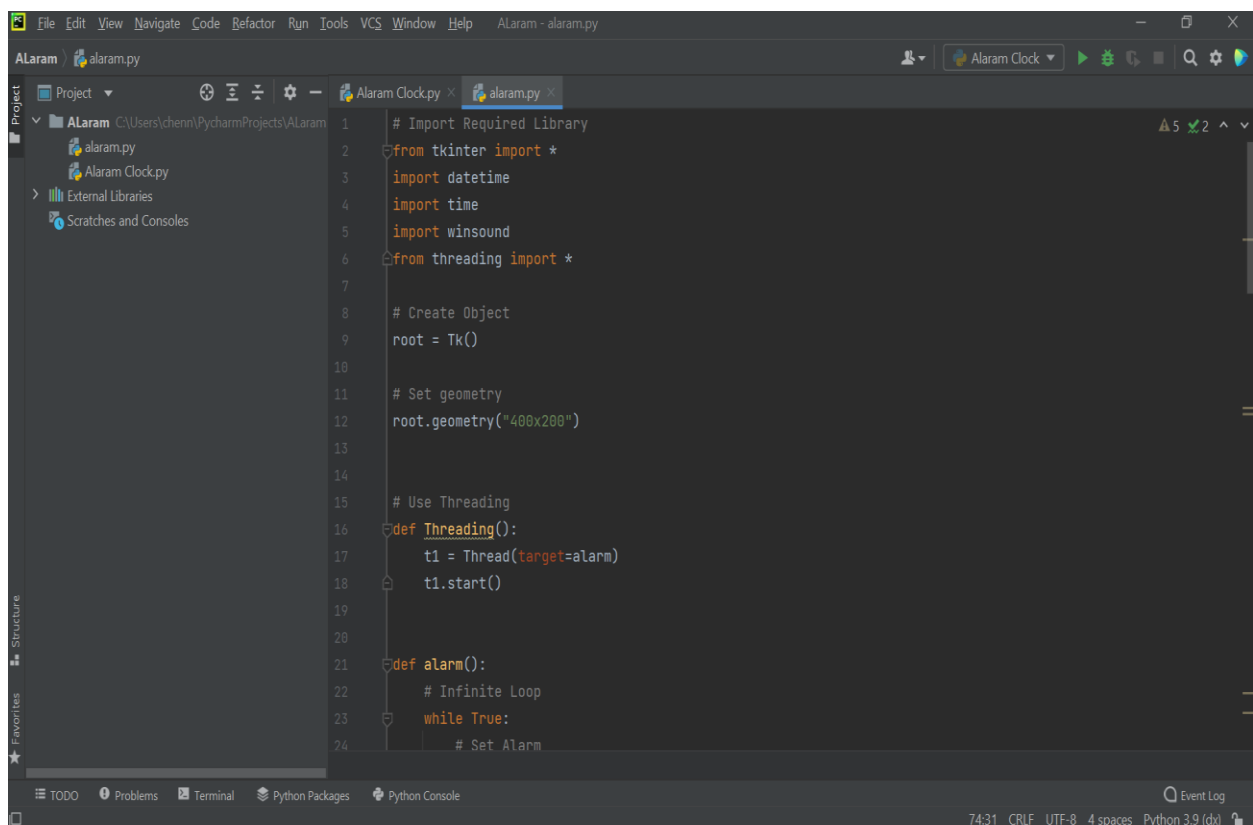
Alarm Clock Using Python- Tkinter project is a desktop application which is developed in Python platform. This Python project with tutorial and guide for developing a code. Alarm Clock Using Python- Tkinter is a open source you can Download zip and edit as per you need. If you want more latest Python projects here. This is simple and basic level small project for learning purpose. Also you can modified this system as per your requirments and develop a perfect advance level project. Zip file containing the source code that can be extracted and then imported into Python IDLE.

Tkinter module belongs to a standard library of GUI in Python. It helps us to create a dialog box with any information that we want to provide or get from the users.

Datetime and time modules in python help us to work with the dates and time of the current day when the user is operating python and to manipulate it too.

Win sound module provides access to the basic sound playing machinery provided by Windows platforms. This is useful to generate the sound immediately when a function is called.

Coding and screen shots of the project



The screenshot displays the PyCharm IDE interface for a project named 'ALaram'. The left sidebar shows the project structure with files 'alarm.py' and 'Alarm Clock.py'. The main editor window shows the code for 'alarm.py'.

```
1 # Import Required Library
2 from tkinter import *
3 import datetime
4 import time
5 import winsound
6 from threading import *
7
8 # Create Object
9 root = Tk()
10
11 # Set geometry
12 root.geometry("400x200")
13
14 # Use Threading
15
16 def Threading():
17     t1 = Thread(target=alarm)
18     t1.start()
19
20
21 def alarm():
22     # Infinite Loop
23     while True:
24         # Set Alarm
```

The bottom status bar indicates the file encoding is UTF-8, using 4 spaces for indentation, and the Python version is 3.9 (dx).

This screenshot shows the initial code in the `alarm.py` file. The code is organized into two main sections: a Tkinter GUI and an infinite loop for alarm functionality.

```
22 # Infinite Loop
23 while True:
24     # Set Alarm
25     set_alarm_time = f"{hour.get()}:{minute.get()}:{second.get()}"
26
27     # Wait for one seconds
28     time.sleep(1)
29
30     # Get current time
31     current_time = datetime.datetime.now().strftime("%H:%M:%S")
32     print(current_time, set_alarm_time)
33
34     # Check whether set alarm is equal to current time or not
35     if current_time == set_alarm_time:
36         print("Time to Wake up")
37         # Playing sound
38         winsound.PlaySound("sound.wav", winsound.SND_ASYNC)
39
40 # Add Labels, Frame, Button, Optionmenus
41 Label(root, text="Alarm Clock", font=("Helvetica 20 bold"), fg="red").pack(pady=10)
42 Label(root, text="Set Time", font=("Helvetica 15 bold")).pack()
43
44
45 frame = Frame(root)
46 alarm()
47 while True:
```

The interface includes a Project view on the left showing the file structure, and a bottom toolbar with tabs for TODO, Problems, Terminal, Python Packages, and Python Console.

This screenshot shows the updated code in the `alarm.py` file, which now includes the Tkinter GUI components. The code is organized into two main sections: GUI setup and the infinite loop.

```
40 # Add Labels, Frame, Button, Optionmenus
41 Label(root, text="Alarm Clock", font=("Helvetica 20 bold"), fg="red").pack(pady=10)
42 Label(root, text="Set Time", font=("Helvetica 15 bold")).pack()
43
44
45 frame = Frame(root)
46 frame.pack()
47
48 hour = StringVar(root)
49 hours = ('00', '01', '02', '03', '04', '05', '06', '07',
50         '08', '09', '10', '11', '12', '13', '14', '15',
51         '16', '17', '18', '19', '20', '21', '22', '23', '24'
52         )
53 hour.set(hours[0])
54
55 hrs = OptionMenu(frame, hour, *hours)
56 hrs.pack(side=LEFT)
57
58 minute = StringVar(root)
59 minutes = ('00', '01', '02', '03', '04', '05', '06', '07',
60           '08', '09', '10', '11', '12', '13', '14', '15',
61           '16', '17', '18', '19', '20', '21', '22', '23',
62           '24', '25', '26', '27', '28', '29', '30', '31',
63           '32', '33', '34', '35', '36', '37', '38', '39'
64           )
65 minute.set(minutes[0])
66
67 alarm()
68 while True:
```

The interface includes a Project view on the left showing the file structure, and a bottom toolbar with tabs for TODO, Problems, Terminal, Python Packages, and Python Console.

```
File Edit View Navigate Code Refactor Run Tools VCS Window Help Alaram - alarm.py
Alaram alarm.py
Project
  Alaram C:\Users\chenn\PycharmProjects\Alaram
    alarm.py
    Alaram Clock.py
  External Libraries
  Scratches and Consoles
Structure
  Alaram
    alarm.py
    Alaram Clock.py
Favorites
64 '40', '41', '42', '43', '44', '45', '46', '47',
65 '48', '49', '50', '51', '52', '53', '54', '55',
66 '56', '57', '58', '59', '60')
67 minute.set(minutes[0])
68
69 mins = OptionMenu(frame, minute, *minutes)
70 mins.pack(side=LEFT)
71
72 second = StringVar(root)
73 seconds = ('00', '01', '02', '03', '04', '05', '06', '07',
74 '08', '09', '10', '11', '12', '13', '14', '15',
75 '16', '17', '18', '19', '20', '21', '22', '23',
76 '24', '25', '26', '27', '28', '29', '30', '31',
77 '32', '33', '34', '35', '36', '37', '38', '39',
78 '40', '41', '42', '43', '44', '45', '46', '47',
79 '48', '49', '50', '51', '52', '53', '54', '55',
80 '56', '57', '58', '59', '60')
81 second.set(seconds[0])
82
83 secs = OptionMenu(frame, second, *seconds)
84 secs.pack(side=LEFT)
85
86 Button(root, text="Set Alarm", font=("Helvetica 15"), command=Threading).pack(pady=20)
87
68:1 CRLF UTF-8 4 spaces Python 3.9 (dx) Event Log
```

```
File Edit View Navigate Code Refactor Run Tools VCS Window Help Alaram - alarm.py
Alaram alarm.py
Project
  Alaram C:\Users\chenn\PycharmProjects\Alaram
    alarm.py
    Alaram Clock.py
  External Libraries
  Scratches and Consoles
Structure
  Alaram
    alarm.py
    Alaram Clock.py
Favorites
71
72 second = StringVar(root)
73 seconds = ('00', '01', '02', '03', '04', '05', '06', '07',
74 '08', '09', '10', '11', '12', '13', '14', '15',
75 '16', '17', '18', '19', '20', '21', '22', '23',
76 '24', '25', '26', '27', '28', '29', '30', '31',
77 '32', '33', '34', '35', '36', '37', '38', '39',
78 '40', '41', '42', '43', '44', '45', '46', '47',
79 '48', '49', '50', '51', '52', '53', '54', '55',
80 '56', '57', '58', '59', '60')
81 second.set(seconds[0])
82
83 secs = OptionMenu(frame, second, *seconds)
84 secs.pack(side=LEFT)
85
86 Button(root, text="Set Alarm", font=("Helvetica 15"), command=Threading).pack(pady=20)
87
88 # Execute Tkinter
89 root.mainloop()
68:1 CRLF UTF-8 4 spaces Python 3.9 (dx) Event Log
```

Conclusion

This project requires good knowledge of Python and GUI (Graphic User Interface). Python when combined with Tkinter provides a fast and easy way to create GUI applications. Tkinter provides a powerful object-oriented interface to the Tk GUI toolkit. All the modules used need not be downloaded beforehand like the other libraries like NumPy, thus this project will be user friendly and accessible in any virtual environment used for python programming.



Reference:

Internet source

Github links:

1. Divya jyothi: <https://github.com/Divyajyothi54>
2. Sai sandhya: <https://github.com/chennurusaisandhya>