

**Internship Report**

Name: Holkar Kalyani Suryakant

Domain name: Python Development

Duration: 1 Month

Task Name : Quiz .

Task Performed : 2

# Task 1 : Quiz Game

**Introduction :**

The Quiz Program is an interactive Python application designed to test the user's knowledge of Python programming. It presents a series of multiple-choice questions related to Python concepts and syntax. The program aims to provide an engaging and educational experience for users, helping them reinforce their Python skills while having fun.

In this quiz program user can interact easily . In this knowledge of python is increased.

Here I use GUI which help user to interact easily with game ,here some radio button ,button ,and background color added for good interaction.

Purpose:

The primary purpose of the Quiz Program is to create an interactive and enjoyable way for users to assess their proficiency in Python. Whether the user is a beginner seeking to solidify their understanding of basic concepts or an experienced developer looking to challenge their advanced knowledge, the quiz caters to all skill levels. Through a diverse range of questions covering various Python topics, users can identify areas they excel in and areas that require further study.

# Task no.1

Quiz Game

**Features:**

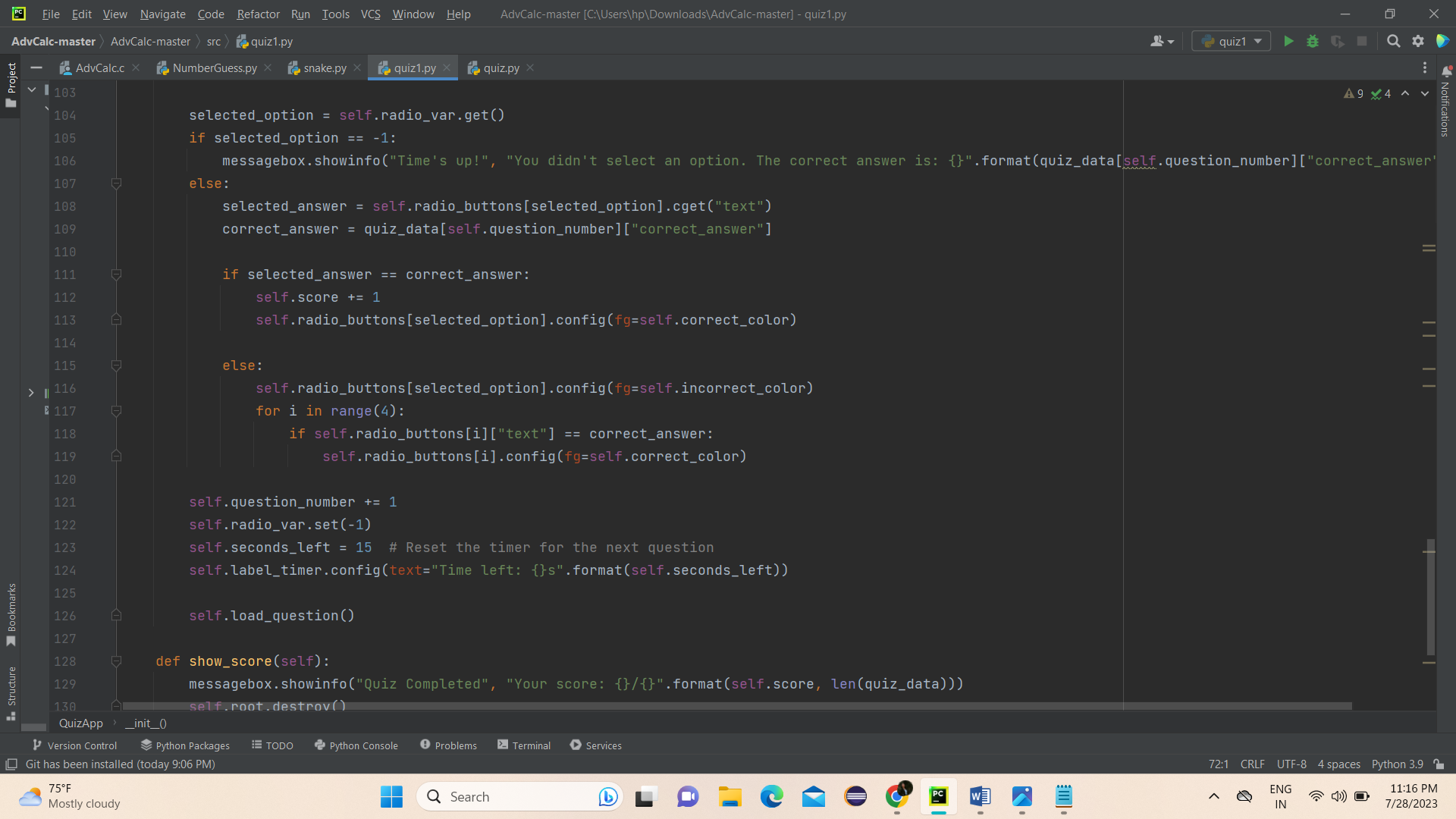
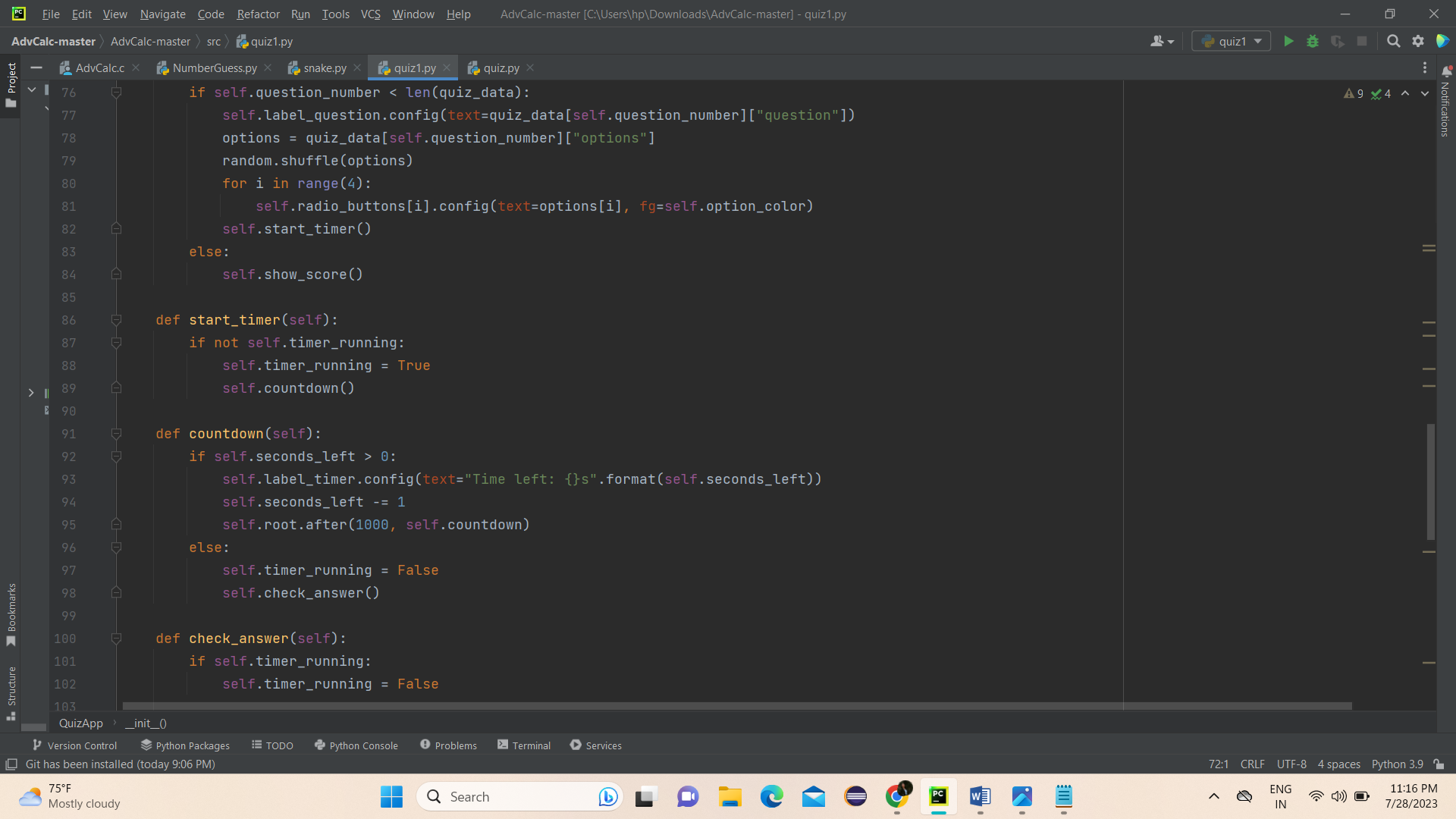
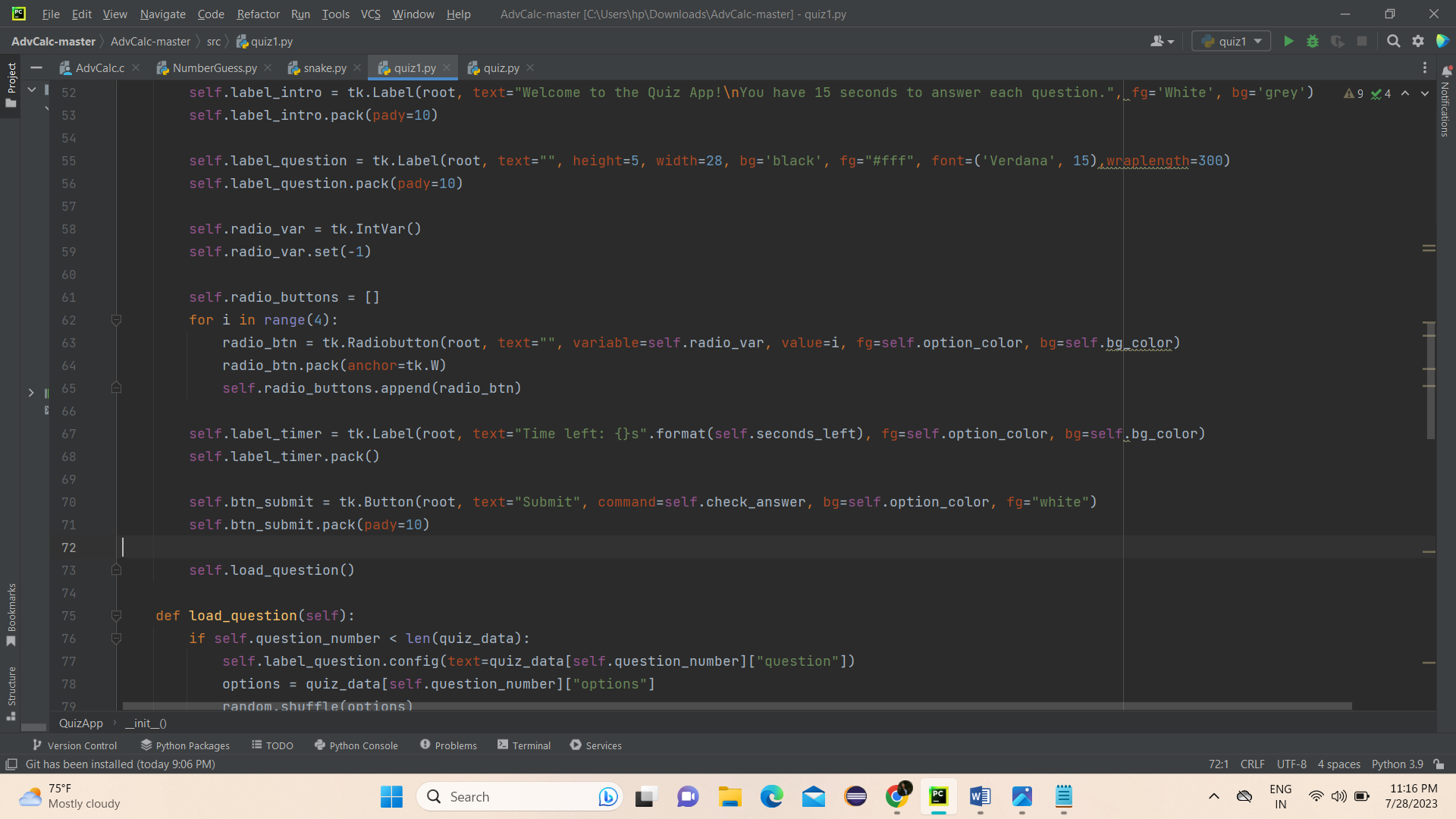
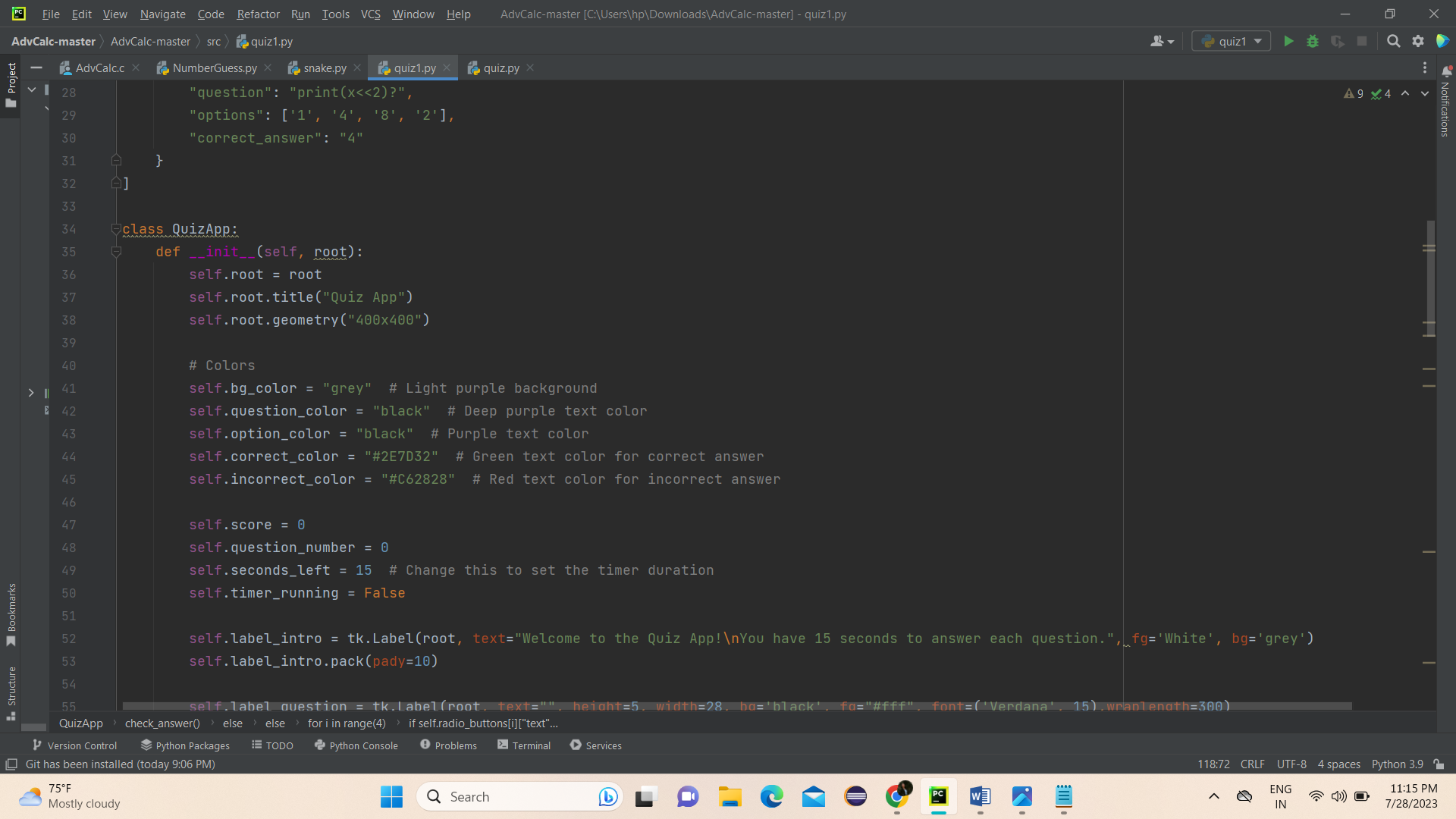
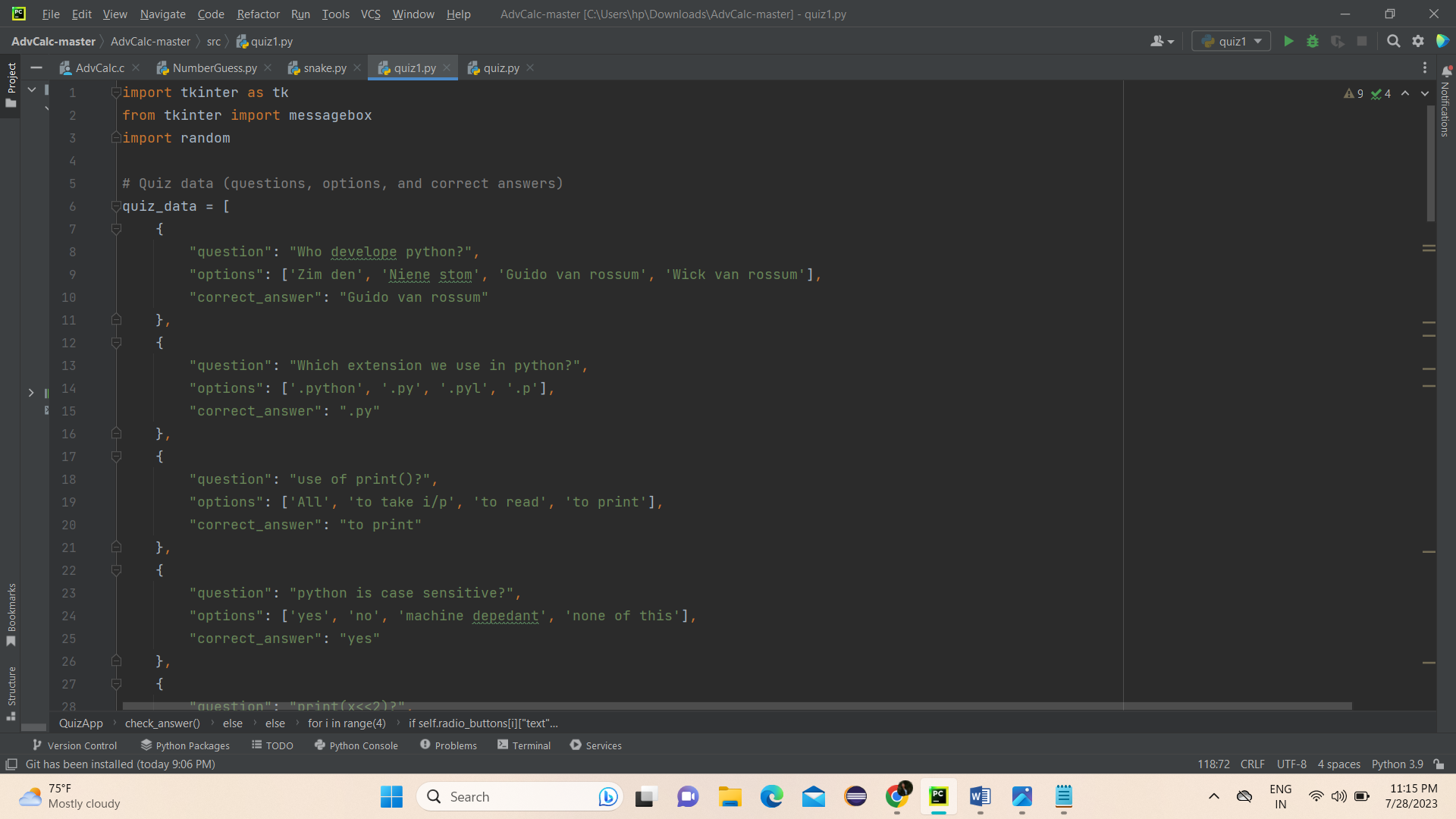
1. Multiple-choice questions: The program includes a diverse set of Python-related questions with four multiple-choice options for each question. This format allows users to select the most appropriate answer from the available options.
2. Interactive User Interface: The quiz is presented through a user-friendly graphical interface that displays questions and provides an intuitive way for users to submit their answers.
3. Scoring System: To motivate users and track their progress, the program implements a scoring system. Each correct answer earns the user a point, and their final score is displayed at the end of the quiz.
4. Timer Functionality: For those seeking an additional challenge, the program can include a timer for each question or for the entire quiz. Users can test their ability to think quickly and make decisions under pressure.

The Quiz Program is implemented in Python. The program utilizes essential components such as lists or dictionaries to store questions and their corresponding answers. The graphical user interface is created using Python libraries like Tkinter, which provide cross-platform support and ease of development.

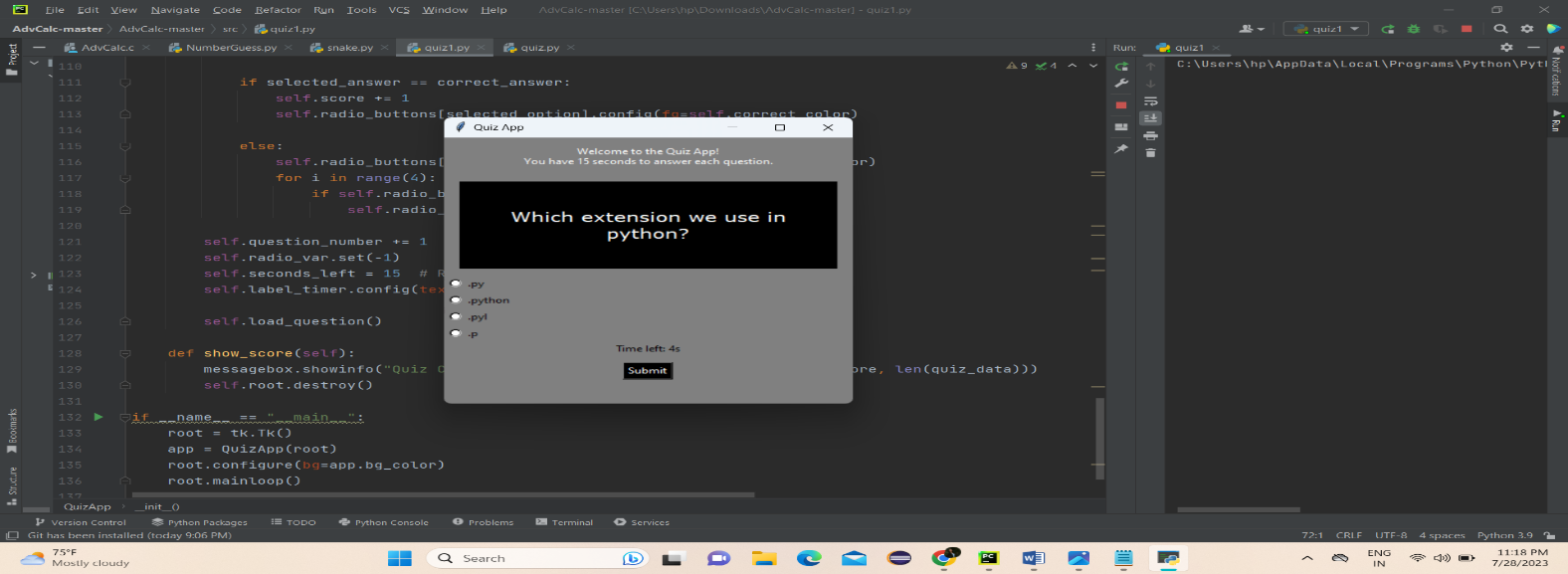
The scoring logic involves tracking correct answers and calculating the user's final score based on their performance throughout the quiz. Timer functionality, if included, is achieved using Python's time module to measure the time taken for each question or the entire quiz.

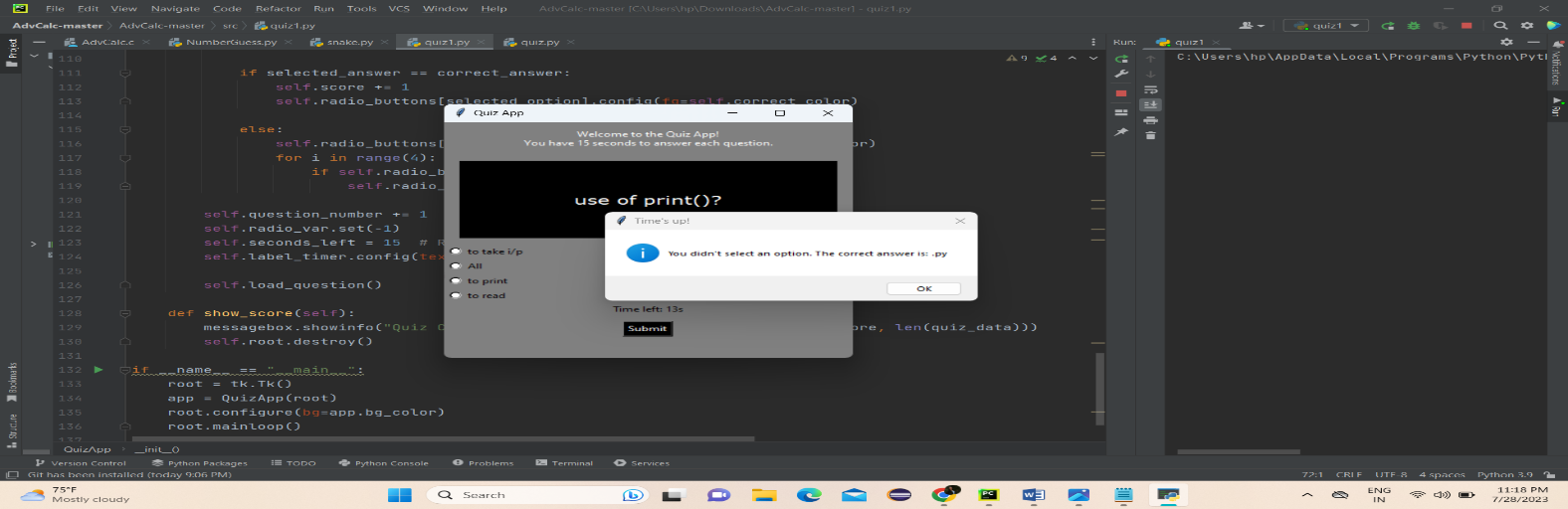
Code :

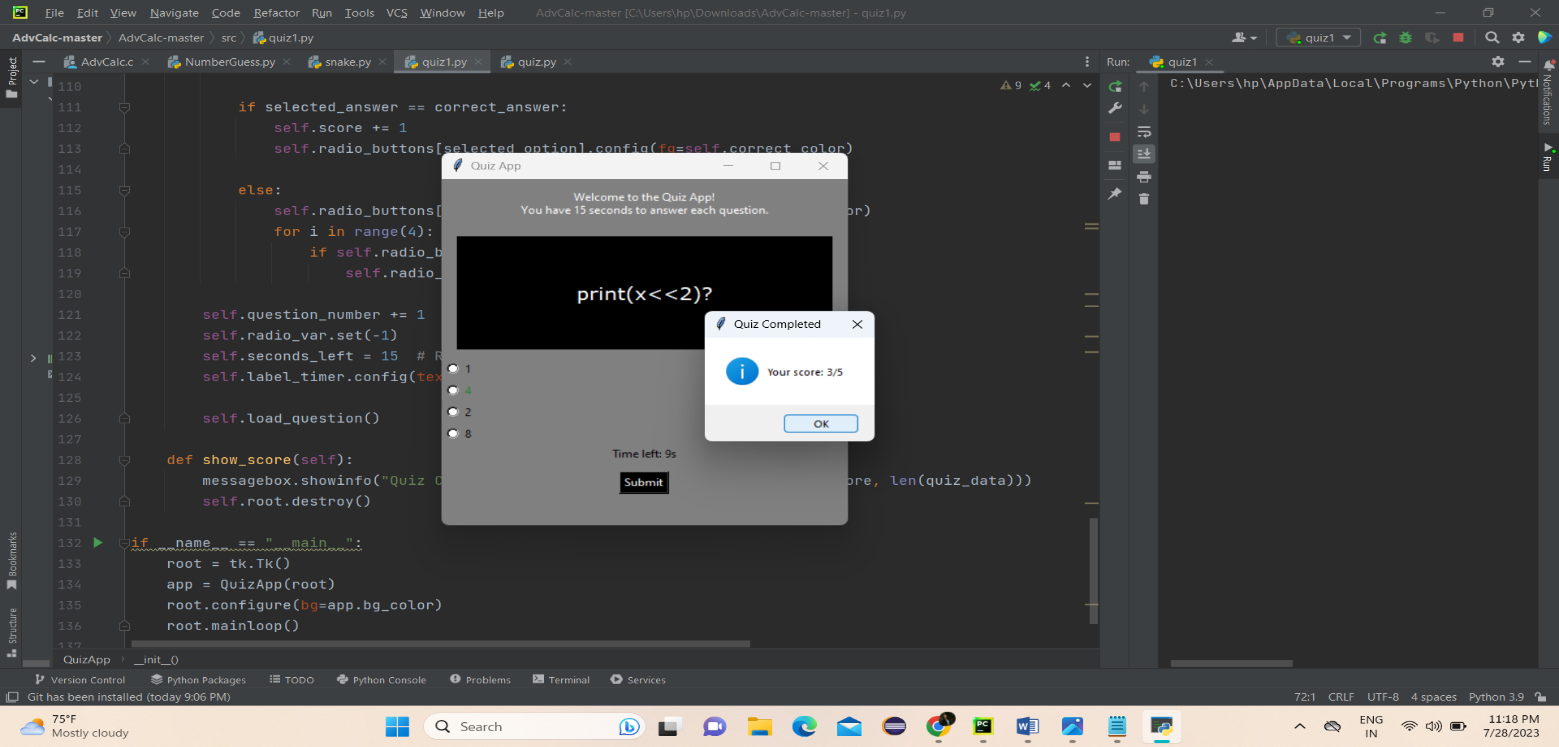
import tkinter as tk  
from tkinter import messagebox  
import random  
  
# Quiz data (questions, options, and correct answers)  
quiz\_data = [  
 {  
 "question": "Who develope python?",  
 "options": ['Zim den', 'Niene stom', 'Guido van rossum', 'Wick van rossum'],  
 "correct\_answer": "Guido van rossum"  
 },  
 {  
 "question": "Which extension we use in python?",  
 "options": ['.python', '.py', '.pyl', '.p'],  
 "correct\_answer": ".py"  
 },  
 {  
 "question": "use of print()?",  
 "options": ['All', 'to take i/p', 'to read', 'to print'],  
 "correct\_answer": "to print"  
 },  
 {  
 "question": "python is case sensitive?",  
 "options": ['yes', 'no', 'machine depedant', 'none of this'],  
 "correct\_answer": "yes"  
 },  
 {  
 "question": "print(x<<2)?",  
 "options": ['1', '4', '8', '2'],  
 "correct\_answer": "4"  
 }  
]  
  
class QuizApp:  
 def \_\_init\_\_(self, root):  
 self.root = root  
 self.root.title("Quiz App")  
 self.root.geometry("400x400")  
  
 # Colors  
 self.bg\_color = "grey" # Light purple background  
 self.question\_color = "black" # Deep purple text color  
 self.option\_color = "black" # Purple text color  
 self.correct\_color = "#2E7D32" # Green text color for correct answer  
 self.incorrect\_color = "#C62828" # Red text color for incorrect answer  
  
 self.score = 0  
 self.question\_number = 0  
 self.seconds\_left = 15 # Change this to set the timer duration  
 self.timer\_running = False  
  
 self.label\_intro = tk.Label(root, text="Welcome to the Quiz App!\nYou have 15 seconds to answer each question.", fg='White', bg='grey')  
 self.label\_intro.pack(pady=10)  
  
 self.label\_question = tk.Label(root, text="", height=5, width=28, bg='black', fg="#fff", font=('Verdana', 15),wraplength=300)  
 self.label\_question.pack(pady=10)  
  
 self.radio\_var = tk.IntVar()  
 self.radio\_var.set(-1)  
  
 self.radio\_buttons = []  
 for i in range(4):  
 radio\_btn = tk.Radiobutton(root, text="", variable=self.radio\_var, value=i, fg=self.option\_color, bg=self.bg\_color)  
 radio\_btn.pack(anchor=tk.W)  
 self.radio\_buttons.append(radio\_btn)  
  
 self.label\_timer = tk.Label(root, text="Time left: {}s".format(self.seconds\_left), fg=self.option\_color, bg=self.bg\_color)  
 self.label\_timer.pack()  
  
 self.btn\_submit = tk.Button(root, text="Submit", command=self.check\_answer, bg=self.option\_color, fg="white")  
 self.btn\_submit.pack(pady=10)  
  
 self.load\_question()  
  
 def load\_question(self):  
 if self.question\_number < len(quiz\_data):  
 self.label\_question.config(text=quiz\_data[self.question\_number]["question"])  
 options = quiz\_data[self.question\_number]["options"]  
 random.shuffle(options)  
 for i in range(4):  
 self.radio\_buttons[i].config(text=options[i], fg=self.option\_color)  
 self.start\_timer()  
 else:  
 self.show\_score()  
  
 def start\_timer(self):  
 if not self.timer\_running:  
 self.timer\_running = True  
 self.countdown()  
  
 def countdown(self):  
 if self.seconds\_left > 0:  
 self.label\_timer.config(text="Time left: {}s".format(self.seconds\_left))  
 self.seconds\_left -= 1  
 self.root.after(1000, self.countdown)  
 else:  
 self.timer\_running = False  
 self.check\_answer()  
  
 def check\_answer(self):  
 if self.timer\_running:  
 self.timer\_running = False  
  
 selected\_option = self.radio\_var.get()  
 if selected\_option == -1:  
 messagebox.showinfo("Time's up!", "You didn't select an option. The correct answer is: {}".format(quiz\_data[self.question\_number]["correct\_answer"]))  
 else:  
 selected\_answer = self.radio\_buttons[selected\_option].cget("text")  
 correct\_answer = quiz\_data[self.question\_number]["correct\_answer"]  
  
 if selected\_answer == correct\_answer:  
 self.score += 1  
 self.radio\_buttons[selected\_option].config(fg=self.correct\_color)  
  
 else:  
 self.radio\_buttons[selected\_option].config(fg=self.incorrect\_color)  
 for i in range(4):  
 if self.radio\_buttons[i]["text"] == correct\_answer:  
 self.radio\_buttons[i].config(fg=self.correct\_color)  
  
 self.question\_number += 1  
 self.radio\_var.set(-1)  
 self.seconds\_left = 15 # Reset the timer for the next question  
 self.label\_timer.config(text="Time left: {}s".format(self.seconds\_left))  
  
 self.load\_question()  
  
 def show\_score(self):  
 messagebox.showinfo("Quiz Completed", "Your score: {}/{}".format(self.score, len(quiz\_data)))  
 self.root.destroy()  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 root = tk.Tk()  
 app = QuizApp(root)  
 root.configure(bg=app.bg\_color)  
 root.mainloop()



Output :







Conclusion :

In conclusion, the quiz game project was an exciting and challenging experience that helped me to increase my python development skills. Through the project, I learned how to use several Gui concept , use of radio button, button, how to set timer and timer module, most interesting thing to creates that thing just we use to play but first time creating is interesting.

Overall this project is very helpful for me.