Objectives:

* If, else if, and else statements.
* Using functions.
* Using Tkinter for GUI apps
* Using while and for loops.

**There are 4 challenge exercise projects, each is worth 25%.**

**Project #1**

In this code, we define a function get\_grade() that takes a score as input and returns the corresponding grade based on the score. The if statements inside the function check different score ranges and assign the appropriate grade.

We then ask the user to enter a score, and we validate that it is within the valid range of 0 to 100. If it is valid, we call the get\_grade() function with the entered score and print the grade. If the score is invalid, we display an error message.

You can run this code and test it by entering different scores to see the corresponding grades.

A screen shot of a computer program

Description automatically generated with medium confidence

**Challenge Exercise #1: complete the program to accept the letter grades of c, d, and f.**

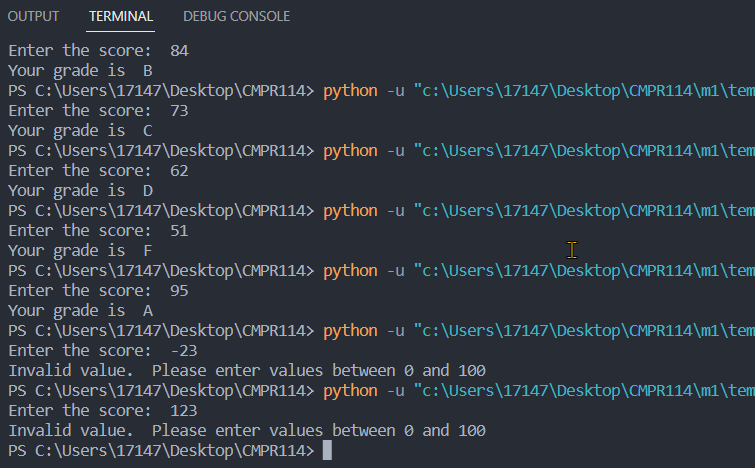
**C = 70-79**

**D = 60-69**

**F = under 60**

**Write a statement if the score is over 100 or a negative number and print the necessary warning.**

**#1 Print screen the running application for all possible combinations with the code, multiple print screens are ok.**



Code:

def get\_grade(score):

if score >= 90 and score < 100:

return "A"

elif score >= 80 and score < 90:

return "B"

elif score >= 70 and score < 80:

return "C"

elif score >= 60 and score < 70:

return "D"

else:

return "F"

def checkinput(score):

if score < 0 or score > 100:

print("Invalid value. Please enter values between 0 and 100")

return False

else:

return True

score = int(input("Enter the score: "))

result = checkinput(score)

if result == True:

lettergrade = get\_grade(score)

print("Your grade is ", lettergrade)

**Project #2**

(Comparing Strings)

Text

Description automatically generatedText

Description automatically generated

**Project #2 (**Using the nested if statement)

Text

Description automatically generated

**Project #3**

In this example, the program prompts the user to enter a number. If the entered number is 0, the program will print "Exiting the program." and break out of the while loop. Otherwise, it checks if the number is even or odd using an if statement with the modulo operator (%). If the number is divisible by 2 (i.e., the remainder is 0), it prints "The number is even." Otherwise, it prints "The number is odd."

The program will continue to prompt for numbers and display whether they are even or odd until the user enters 0 to quit.

A picture containing text, screenshot, display, font

Description automatically generated

**Project #4**

In this example, the program initializes an empty list called students. Inside the while loop, the user is prompted to enter a student’s name. If the user enters "quit" (case-insensitive), the loop breaks, and the program moves to the next part. Otherwise, the entered name is appended to the student’s list.

After the loop finishes, the program prints all the student names stored in the list using a for loop.

The program will continue to ask for student names until the user enters "quit" to stop.

A picture containing text, screenshot, font

Description automatically generated

**Project #5**

In this example, we create a **GUI** window using Tkinter. Three input fields are provided for the user to enter the last name, first name, and address. The submit function is called when the "Submit" button is clicked.

Inside the submit function, the values from the input fields are retrieved using the .get() method. An if statement checks if all the fields have been filled in. If they have, the function prints the last name, first name, and address. If any field is missing, it prints a message asking the user to fill in all the fields.

You can run this code to see the Tkinter window with the input fields. When the "Submit" button is clicked, the entered values will be printed if all the fields are filled. Otherwise, an error message will be displayed.

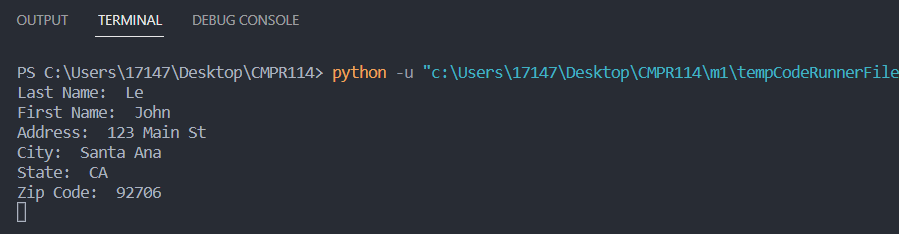
A picture containing text, screenshot, software, font

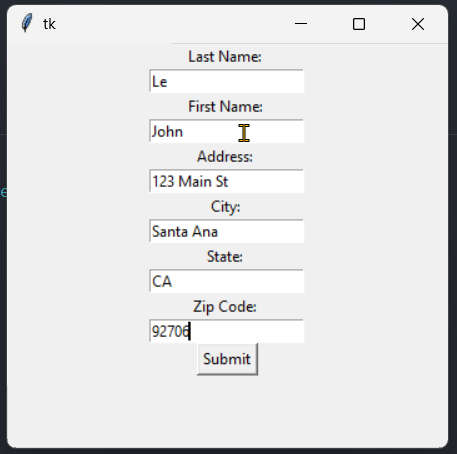
Description automatically generatedA picture containing text, screenshot, font

Description automatically generated

**Challenge Exercise #2 Complete the above program and add the city, and state, with zip code.**

**#2 Print screen the output with the code below here**

****

****

Code:

import tkinter as tk

#Project 2 -Complete existing code

def submit():

lastname = entry\_lastname.get()

firstname = entry\_firstname.get()

address = entry\_address.get()

city = entry\_city.get()

state = entry\_state.get()

zipcode = entry\_zipcode.get()

if lastname and firstname and address and city and state and zipcode:

print("Last Name: ", lastname)

print("First Name: ", firstname)

print("Address: ", address)

print("City: ", city)

print("State: ", state)

print("Zip Code: ", zipcode)

else:

print("Please fill in all fields.")

window = tk.Tk()

label\_lastname = tk.Label(window, text = "Last Name: ")

label\_lastname.pack()

entry\_lastname = tk.Entry(window)

entry\_lastname.pack()

label\_firstname = tk.Label(window, text = "First Name: ")

label\_firstname.pack()

entry\_firstname = tk.Entry(window)

entry\_firstname.pack()

label\_address = tk.Label(window, text = "Address: ")

label\_address.pack()

entry\_address = tk.Entry(window)

entry\_address.pack()

label\_city = tk.Label(window, text = "City: ")

label\_city.pack()

entry\_city = tk.Entry(window)

entry\_city.pack()

label\_state = tk.Label(window, text = "State: ")

label\_state.pack()

entry\_state = tk.Entry(window)

entry\_state.pack()

label\_zipcode = tk.Label(window, text = "Zip Code: ")

label\_zipcode.pack()

entry\_zipcode = tk.Entry(window)

entry\_zipcode.pack()

submit\_button = tk.Button(window, text="Submit", command=submit)

submit\_button.pack()

window.mainloop()

**Project #6**

This GUI app will submit a function that opens a file named "student\_data.txt" in append mode using the with open() statement. The function then writes the last name, first name, and address to the file, separated by newlines. After writing the data, it prints a message indicating that the data has been saved.

You can run this code, enter the student information in the Tkinter window, and click the "Submit" button. The entered data will be saved to the "student\_data.txt" file in the same directory as your Python script. Each time you submit new data, it will be appended to the file on a new line.

A picture containing text, screenshot, font

Description automatically generated

A screen shot of a computer program

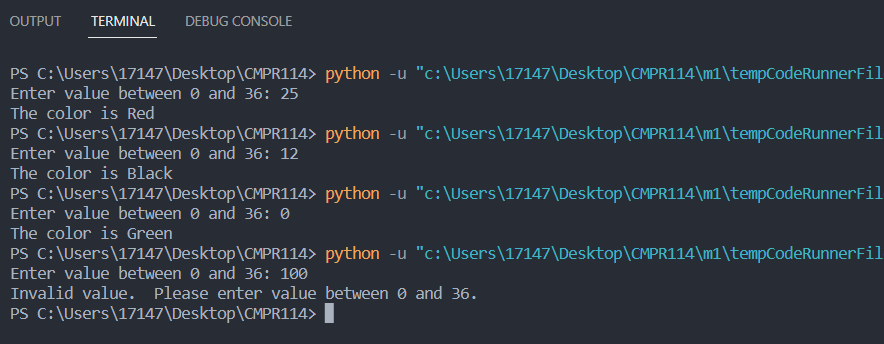
Description automatically generated with low confidence

**Challenge Exercise #3** Complete the following program using the if block structure.

Text

Description automatically generated with medium confidence

**#3 Print screen the output with the code below here**



Code:

def validate(val):

if val < 0 or val > 36:

print("Invalid value. Please enter value between 0 and 36.")

return False

else:

return True

def get\_numtype(val):

remainder = val % 2

if remainder > 0:

numtype = "odd"

else:

numtype = "even"

def get\_color(val):

numtype = get\_numtype(val)

if val == 0:

return "Green"

elif val >= 1 and val < 11:

if numtype == "even":

return "Black"

else:

return "Red"

elif val >= 11 and val < 19:

if numtype == "even":

return "Red"

else:

return "Black"

elif val >= 19 and val < 28:

if numtype == "even":

return "Black"

else:

return "Red"

elif val >= 29 and val < 37:

if numtype == "even":

return "Red"

else:

return "Black"

val = int(input("Enter value between 0 and 36: "))

result = validate(val)

if result == True:

color = get\_color(val)

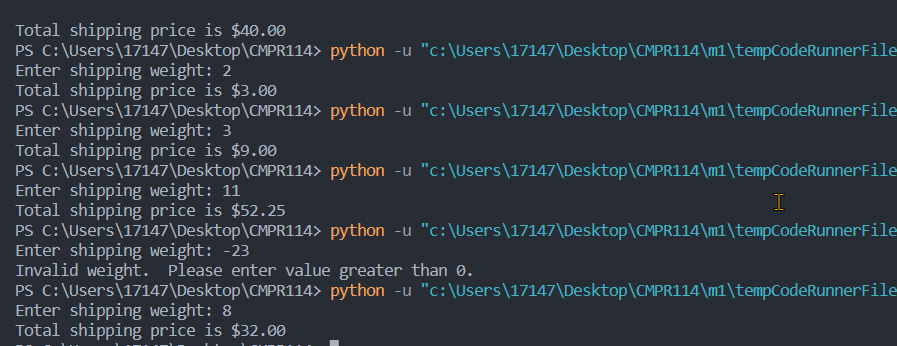
print("The color is ", color, sep = "")

**Challenge Exercise #4:** complete the following program using the if block structure.

Graphical user interface, text, timeline

Description automatically generated

**#4 Print screen the output with the code below here**

****

def validate(weight):

if weight < 0:

print("Invalid weight. Please enter value greater than 0.")

return False

else:

return True

def calculate\_price(weight):

if weight > 0 and weight <= 2:

price = weight \* 1.5

elif weight > 2 and weight <= 6:

price = weight \* 3.0

elif weight > 6 and weight <= 10:

price = weight \* 4

else:

price = weight \* 4.75

return price

weight = float(input("Enter shipping weight: "))

result = validate(weight)

if result == True:

shippingprice = calculate\_price(weight)

print("Total shipping price is ${:,.2f}".format(shippingprice))

**Submit this document to the Module 1 exercise.**