Task Level (Beginner)

1. Variables

1. Create a variable named pi and store the value 22/7 in it. Now check the data type of this variable.

CODE:  
pi=22/7 ##outpt: 3.142857142857143

print(pi)

The value 3.142857142857143 is a floating-point number, a decimal approximation of the fraction 22/7.

2.Create a variable called for and assign it a value 4. See what happens and find out the reason behind the behaviour that you see.

CODE:

for=5

print(5) output: invalid syntax

as for is a Reserved keywords . Reserved keywords are part of the syntax and have special meanings in

the language, so using them as variable names results in a syntax error.

3.Store the principal amount, rate of interest, and time in different variables and then calculate the Simple Interest for 3 years. Formula: Simple Interest = P x R x T / 100

P = 1000 # Principal amount

R = 5 # Rate of interest

T = 3 # Time IN years

# Calculating Simple Interest output: Simple Interest for 3 years is: 150.0

simple\_interest = (P \* R \* T) / 100

# Output the result

print("Simple Interest for 3 years is:", simple\_interest)

2. Numbers

1. Write a function that takes two arguments, 145 and 'o', and uses the `format` function to return a formatted string. Print the result. Try to identify the representation used.

CODE:  
def format\_string(num, representation):

# Using format function to apply the representation

formatted\_string = "{:o}".format(num) # 'o' is for octal representation

return formatted\_string

# Calling function with 145 and 'o' as arguments output: Simple Interest for 3 years is: 150.0

result = format\_string(145, 'o')

print(result)

1. In a village, there is a circular pond with a radius of 84 meters. Calculate the area of the pond using the formula: Circle Area = π r^2. (Use the value 3.14 for π) Bonus Question: If there is exactly 1.4 liters of water in a square meter, what is the total amount of water in the pond? Print the answer without any decimal point in it. Hint: Circle Area = π r^2 Water in the pond = Pond Area Water per Square Meter

# Given values

pi = 3.14

radius = 84

water\_per\_square\_meter = 1.4

OUTPUT:

# Calculating the area of the pond The area of the pond is approximately 22176.96 square meters

area\_of\_pond = pi \* radius \*\* 2 The total amount of water in the pond is 31047 liters

# Calculating the total amount of water in the pond

total\_water = area\_of\_pond \* water\_per\_square\_meter

print("The area of the pond is approximately:", round(area\_of\_pond, 2), "square meters.")

print("The total amount of water in the pond is:", int(total\_water), "liters.")

3.If you cross a 490meterlong street in 7 minutes, calculate your speed in meters per second. Print the answer without any decimal point in it. Hint: Speed = Distance / Time

# Given values

distance = 490 # meters

time\_minutes = 7 # minutes

# Converting time to seconds

time\_seconds = time\_minutes \* 60

# Calculating speed

speed = distance / time\_seconds #formule

print(int(speed)) # given int to print without decimal

3. List

1. You have a list of superheroes representing the Justice League. justice\_league = ["Superman", "Batman", "Wonder Woman", "Flash", "Aquaman", "Green Lantern"]

1. Calculate the number of members in the Justice League.  
CODE:  
justice\_league = ["Superman", "Batman", "Wonder Woman", "Flash", "Aquaman", "Green Lantern"]

print("The number of members in the Justice League:", len(justice\_league))

output:  
The number of members in the Justice League: 6

2.Batman recruited Batgirl and Nightwing as new members. Add them to your list.

CODE:  
justice\_league = ["Superman", "Batman", "Wonder Woman", "Flash", "Aquaman", "Green Lantern"]

new\_recruit=input().split()

justice\_league.append(new\_recruit)

print(justice\_league)

output:  
Batgirl Nightwing

['Superman', 'Batman', 'Wonder Woman', 'Flash', 'Aquaman', 'Green Lantern', ['Batgirl', 'Nightwing']]

3. Wonder Woman is now the leader of the Justice League. Move her to the beginning of the list.  
CODE:  
justice\_league = ['Superman', 'Batman', 'Wonder Woman', 'Flash', 'Aquaman', 'Green Lantern', ['Batgirl', 'Nightwing']]

wonder\_woman\_index = justice\_league.index('Wonder Woman')

# Swap Wonder Woman with the first element

justice\_league[wonder\_woman\_index], justice\_league[0] = justice\_league[0], justice\_league[wonder\_woman\_index]

print("Updated Justice League:", justice\_league)  
output:

Updated Justice League: ['Wonder Woman', 'Batman', 'Superman', 'Flash', 'Aquaman', 'Green Lantern', ['Batgirl', 'Nightwing']]

4.Aquaman and Flash are having conflicts, and you need to separate them. Choose either "Green Lantern" or "Superman" and move them in between Aquaman and Flash.

CODE:  
justice\_league = ['Superman', 'Batman', 'Wonder Woman', 'Flash', 'Aquaman', 'Green Lantern', ['Batgirl', 'Nightwing']]

# Find indices of Aquaman and Flash

aquaman\_index = justice\_league.index('Aquaman')

flash\_index = justice\_league.index('Flash')

# Remove 'Green Lantern' and insert them between Aquaman and Flash

green\_lantern = justice\_league.pop(justice\_league.index('Green Lantern'))

justice\_league.insert(flash\_index, green\_lantern)

print("Updated Justice League:", justice\_league)

out put:  
Updated Justice League: ['Superman', 'Batman', 'Wonder Woman', 'Aquaman', 'Green Lantern', 'Flash', ['Batgirl', 'Nightwing']]

5. . The Justice League faced a crisis, and Superman decided to assemble a new team. Replace the existing list with the following new members: "Cyborg", "Shazam", "Hawkgirl", "Martian Manhunter", "Green Arrow".  
CODE:  
justice\_league = ['Superman', 'Batman', 'Wonder Woman', 'Flash', 'Aquaman', 'Green Lantern', ['Batgirl', 'Nightwing']]

# Replace the existing list with the new members

justice\_league = ["Cyborg", "Shazam", "Hawkgirl", "Martian Manhunter", "Green Arrow"]

print("Updated Justice League:", justice\_league)

output:

Updated Justice League: ['Cyborg', 'Shazam', 'Hawkgirl', 'Martian Manhunter', 'Green Arrow']

6. . Sort the Justice League alphabetically. The hero at the 0th index will become the new leader.

CODE:  
justice\_league = ["Cyborg", "Shazam", "Hawkgirl", "Martian Manhunter", "Green Arrow"]

# Sort the list alphabetically

justice\_league.sort()

# The hero at the 0th index becomes the new leader

new\_leader = justice\_league[0]

print("Sorted Justice League:", justice\_league)

print("The new leader of the Justice League is:", new\_leader)

output:

Sorted Justice League: ['Cyborg', 'Green Arrow', 'Hawkgirl', 'Martian Manhunter', 'Shazam']

The new leader of the Justice League is: Cyborg

(BONUS : I predict **Martian Manhunter** can be the new leader, Martian Manhunter is one of the most experienced and wise members of the Justice League. He’s calm, empathetic, and a strategic thinker. He has a deep sense of responsibility and could lead the team in a balanced and thoughtful way.)

Final code:  
# Initial list of Justice League members

justice\_league = ['Superman', 'Batman', 'Wonder Woman', 'Flash', 'Aquaman', 'Green Lantern', ['Batgirl', 'Nightwing']]

# 1. Move Wonder Woman to the beginning of the list

justice\_league.remove("Wonder Woman")

justice\_league.insert(0, "Wonder Woman")

print("Step 1: Move Wonder Woman to the beginning")

print(justice\_league)

# 2. Separate Aquaman and Flash by inserting Green Lantern between them

aquaman\_index = justice\_league.index('Aquaman')

flash\_index = justice\_league.index('Flash')

# Remove Green Lantern and insert it between Aquaman and Flash

green\_lantern = justice\_league.pop(justice\_league.index('Green Lantern'))

justice\_league.insert(flash\_index, green\_lantern)

print("\nStep 2: Separate Aquaman and Flash by inserting Green Lantern")

print(justice\_league)

# 3. Replace the list with new members

justice\_league = ["Cyborg", "Shazam", "Hawkgirl", "Martian Manhunter", "Green Arrow"]

print("\nStep 3: Replace the existing list with the new members")

print(justice\_league)

# 4. Sort the Justice League alphabetically and find the new leader

justice\_league.sort()

new\_leader = justice\_league[0]

print("\nStep 4: Sorted Justice League and identified new leader")

print("Sorted Justice League:", justice\_league)

print("The new leader of the Justice League is:", new\_leader)

output:  
Step 1: Move Wonder Woman to the beginning

['Wonder Woman', 'Superman', 'Batman', 'Flash', 'Aquaman', 'Green Lantern', ['Batgirl', 'Nightwing']]

Step 2: Separate Aquaman and Flash by inserting Green Lantern

['Wonder Woman', 'Superman', 'Batman', 'Aquaman', 'Green Lantern', 'Flash', ['Batgirl', 'Nightwing']]

Step 3: Replace the existing list with the new members

['Cyborg', 'Shazam', 'Hawkgirl', 'Martian Manhunter', 'Green Arrow']

Step 4: Sorted Justice League and identified new leader

Sorted Justice League: ['Cyborg', 'Green Arrow', 'Hawkgirl', 'Martian Manhunter', 'Shazam']

The new leader of the Justice League is: Cyborg

note: **Cyborg** is the new leader of the Justice League after sorting the list alphabetically.

4. If Condition:  
 1. Write a program to determine the BMI Category based on user input. Ask the user to: Enter height in meters Enter weight in kilograms Calculate BMI using the formula: BMI = weight / (height)2 Use the following categories: If BMI is 30 or greater, print "Obesity" If BMI is between 25 and 29, print "Overweight" If BMI is between 18.5 and 25, print "Normal" If BMI is less than 18.5, print "Underweight"

Example: Enter height in meters: 1.75 ,Enter weight in kilograms: 70

Output: "Normal"

CODE:

# BMI Calculation and Category Determination

# Step 1: Get user input for height and weight

height = float(input("Enter height in meters: "))

weight = float(input("Enter weight in kilograms: ")) inputs: Enter height in meters: 1.75

# Step 2: Calculate BMI using the formula Enter weight in kilograms: 70

bmi = weight / (height \*\* 2)

# Step 3: Determine the BMI category output: Normal

if bmi >= 30:

print("Obesity")

elif 25 <= bmi < 30:

print("Overweight")

elif 18.5 <= bmi < 25:

print("Normal")

else:

print("Underweight")

2. Write a program to determine which country a city belongs to. Given list of cities per country: Australia = ["Sydney", "Melbourne", "Brisbane", "Perth"] UAE = ["Dubai", "Abu Dhabi", "Sharjah", "Ajman"] India = ["Mumbai", "Bangalore", "Chennai", "Delhi"] Ask the user to enter a city name and print the corresponding country. Example: Enter a city name: "Abu Dhabi" Output: "Abu Dhabi is in UAE"

CODE:  
# List of cities per country

Australia = ["Sydney", "Melbourne", "Brisbane", "Perth"]

UAE = ["Dubai", "Abu Dhabi", "Sharjah", "Ajman"]

India = ["Mumbai", "Bangalore", "Chennai", "Delhi"]

# Step 1: Ask the user to enter a city name

city = input("Enter a city name: ") 1.input: Enter a city name: Abu Dhabi

# Step 2: Determine the country of the city output: Abu Dhabi is in UAE

if city in Australia: 2. Enter a city name: Mumbai

print(f"{city} is in Australia") output: Mumbai is in India

elif city in UAE:

print(f"{city} is in UAE")

elif city in India:

print(f"{city} is in India")

else:

print(f"{city} is not in the database of cities.")

1. Write a program to check if two cities belong to the same country. Ask the user to enter two cities and print whether they belong to the same country or not. Example: Enter the first city: "Mumbai" Enter the second city: "Chennai" Output: "Both cities are in India" Example: Enter the first city: "Sydney" Enter the second city: "Dubai" Output: "They don't belong to the same country"

CODE:

Australia = ["Sydney", "Melbourne", "Brisbane", "Perth"]

UAE = ["Dubai", "Abu Dhabi", "Sharjah", "Ajman"]

India = ["Mumbai", "Bangalore", "Chennai", "Delhi"]

# Step 1: Ask the user to enter the first and second city names

city1 = input("Enter the first city: ")

city2 = input("Enter the second city: ")

# Step 2: Check which country each city belongs to

if city1 in Australia and city2 in Australia:

print("Both cities are in Australia")

elif city1 in UAE and city2 in UAE:

print("Both cities are in UAE")

elif city1 in India and city2 in India:

print("Both cities are in India")

else:

print("They don't belong to the same country")

5. For Loop:  
 1. Using a for loop, simulate rolling a six sided die multiple times (at least 20 times). Count and print the following statistics: How many times you rolled a 6 How many times you rolled a 1 How many times you rolled two 6s in a row

Code:  
import random

count\_6 = 0

count\_1 = 0

count\_double\_6 = 0

previous\_roll = None

for i in range(20):

roll = random.randint(1, 6)

if roll == 6:

count\_6 += 1

elif roll == 1:

count\_1 += 1

if roll == 6 and previous\_roll == 6:

count\_double\_6 += 1

previous\_roll = roll

print(f"Times rolled a 6: {count\_6}")

print(f"Times rolled a 1: {count\_1}")

print(f"Times rolled two 6s in a row: {count\_double\_6}")

output:  
Times rolled a 6: 4

Times rolled a 1: 3

Times rolled two 6s in a row: 1

2. Imagine you are doing a workout routine, and you have to complete 100 jumping jacks. Write a program that: Asks you to perform 10 jumping jacks at a time. After each set, it asks, "Are you tired?" If you reply "yes" or "y," it should ask if you want to skip the remaining sets. If you reply "yes" or "y," it should break and print, "You completed a total of jumping jacks."

For example, if you did only 30 jumping jacks and answered "yes," the program will break and print, "You completed a total of 30 jumping jacks." If you reply "no" or "n," it should continue and display how many jumping jacks are remaining. After that, ask you again, "Are you tired?"

For example, if you answered "no," it should display that 70 jumping jacks are remaining and ask you again, "Are you tired?" If you reply "no" or "n," it should continue and display how many jumping jacks are remaining. After that, ask you again, "Are you tired?"

For example, if you answered "no," it should display that 70 jumping jacks are remaining and ask you again, "Are you tired?" If you complete all 100 jumping jacks, it should print, "Congratulations! You completed the workout," and stop the program

Code:

total\_jumping\_jacks = 0

for i in range(10): # 10 sets of 10 jumping jacks

total\_jumping\_jacks += 10

print(f"You've completed {total\_jumping\_jacks} jumping jacks.")

tired = input("Are you tired? (yes/no): ").lower()

if tired == "yes" or tired == "y":

skip = input("Do you want to skip the remaining sets? (yes/no): ").lower()

if skip == "yes" or skip == "y":

print(f"You completed a total of {total\_jumping\_jacks} jumping jacks.")

break

else:

remaining = 100 - total\_jumping\_jacks

if remaining > 0:

print(f"{remaining} jumping jacks remaining.")

else:

print("Congratulations! You completed the workout.")

break

sample output:  
You've completed 10 jumping jacks.

Are you tired? (yes/no):

90 jumping jacks remaining.

You've completed 20 jumping jacks.

Are you tired? (yes/no): y

Do you want to skip the remaining sets? (yes/no): n

You've completed 30 jumping jacks.

Are you tired? (yes/no): y

Do you want to skip the remaining sets? (yes/no): n

You've completed 40 jumping jacks.

Are you tired? (yes/no): y

Do you want to skip the remaining sets? (yes/no): y

You completed a total of 40 jumping jacks.