Rajalakshmi Engineering College

Name: krithika narasimhan
Email: 240701277@rajalakshmi.edu.in
Roll no: 240701277
Phone: 9677451731
Branch: REC
Department: I CSE FC
Batch: 2028
Degree: B.E - CSE

Scan to verify results

NeoColab_REC_CS23221_Python Programming

REC_Python_Week 1_PAH

Attempt : 2
Total Mark : 6
Marks Obtained : 6

Section 1 : Coding

1. Problem Statement

Ella, an avid TV show enthusiast, is planning a binge-watching marathon for a new series. She has a specific routine: after watching a set number of episodes, she takes a short break.

She is provided with the following information:

Each episode of the series has a fixed duration of 45 minutes.After a certain number of episodes, there is a break of 15 minutes.

Ella wants to know the total time she will need to watch the entire series, including the breaks. Your task is to help Ella by calculating the total viewing time.

Input Format

The first line of input consists of an integer E, representing the total number of episodes in the series.

The second line consists of an integer B, representing the number of episodes watched before taking a break.

Output Format

The output prints an integer representing the total viewing time required to watch the entire series, including the breaks.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 5
2
Output: 255 minutes

Answer

```
total_episodes = int(input())
episodes_before_break = int(input())
episode_duration = 45 # in minutes
break_duration = 15   # in minutes

full_breaks = (total_episodes - 1) // episodes_before_break # Calculate full breaks
remaining_episodes = total_episodes - (full_breaks * episodes_before_break) # Calculate remaining episodes

total_duration = (full_breaks * (episodes_before_break * episode_duration + break_duration) + (remaining_episodes * episode_duration)) * (full_breaks > 0) + (total_episodes * episode_duration) * (full_breaks == 0)

print(f"{total_duration} minutes")
```

Status : Correct ***Marks : 1/1***

2. Problem Statement

A smart home system tracks the temperature and humidity of each room. Create a program that takes the room name (string), temperature (float), and humidity (float).

Display the room's climate details.

Input Format

The first line of input consists of a string, representing the room name.

The second line consists of a float value, representing the temperature.

The third line consists of a float value, representing the humidity.

Output Format

The first line of output prints "Room: " followed by the room name (string).

The second line prints "Temperature: " followed by the temperature (float) formatted to two decimal places.

The third line prints "Humidity: " followed by the humidity (float) formatted to two decimal places and a percentage sign (%).

Refer to the sample output for formatting specifications.

Sample Test Case

Input: Living Room
23.45
45.78

Output: Room: Living Room
Temperature: 23.45
Humidity: 45.78%

Answer

```
room_name = input()
temperature = float(input())
humidity = float(input())
```

```
print(f"Room: {room_name}")
print(f"Temperature: {temperature:.2f}")
print(f"Humidity: {humidity:.2f}%")
```

Status : Correct ***Marks : 1/1***

3. Problem Statement

Shawn, a passionate baker, is planning to bake cookies for a large party. His original recipe makes 15 cookies, with the following ingredient quantities: 2.5 cups of flour, 1 cup of sugar, and 0.5 cups of butter.

Write a program to calculate the amounts of flour, sugar, and butter needed for a different number of cookies. Provide the ingredient quantities for a specified number of cookies, maintaining the original proportions of the recipe.

Input Format

The input consists of an integer n, representing the number of cookies.

Output Format

The first line prints "Flour: X cups" where X represents the amount of flour required for n cookies, as a double value rounded to two decimal places.

The second line prints "Sugar: Y cups" where Y represents the amount of Sugar required for n, as a double value rounded to two decimal places.

The third line prints "Butter: Z cups" where Z represents the amount of flour required for n, as a double value rounded to two decimal places.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 15
Output: Flour: 2.50 cups
Sugar: 1.00 cups

Butter: 0.50 cups

Answer

```
x = int(input())

flour_per_12_cupcakes = 2.5
sugar_per_12_cupcakes = 1.0
butter_per_12_cupcakes = 0.5

flour_needed = (x / 15.0) * flour_per_12_cupcakes
sugar_needed = (x / 15.0) * sugar_per_12_cupcakes
butter_needed = (x / 15.0) * butter_per_12_cupcakes

print(f"Flour: {flour_needed:.2f} cups")
print(f"Sugar: {sugar_needed:.2f} cups")
print(f"Butter: {butter_needed:.2f} cups")
```

Status : Correct ***Marks : 1/1***

4. Problem Statement

Mandy is debating with her friend Rachel about an interesting mathematical claim. Rachel asserts that for any positive integer n, the ratio of the sum of n and its triple to the integer itself is always 4. Mandy, intrigued by this statement, decides to validate it using logical operators and basic arithmetic.

She wants to confirm if the statement holds true for any positive integer n.

Input Format

The input consists of a positive integer n, representing the integer to be tested.

Output Format

The first line of output displays "Sum:" followed by an integer representing the calculated sum.

The second line displays "Rachel's statement is: " followed by a Boolean value indicating whether Rachel's statement is correct.

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 12

Output: Sum: 48
Rachel's statement is: True

Answer

```
n = int(input())

sum_with_triple = n + 3 * n

ratio = sum_with_triple / n

# Validate Rachel's statement using a logical comparison
rachel_statement_correct = (ratio == 4)

# Output the result and the validation
print("Sum:", sum_with_triple)
print("Rachel's statement is:", rachel_statement_correct)
```

Status : Correct ***Marks : 1/1***

5. Problem Statement

Oliver is planning a movie night with his friends and wants to download a high-definition movie. He knows the file size of the movie in megabytes (MB) and his internet speed in megabits per second (Mbps). To ensure the movie is ready in time, Oliver needs to calculate the download time.

Your task is to write a program that calculates the download time and displays it in hours, minutes, and seconds.

Example

Input:

MB = 800

mbps = 40

Output:

Download Time: 0 hours, 2 minutes, and 40 seconds

Explanation:

Convert the file size to bits (800 MB * 8 bits/byte = 6400 megabits) and divide it by the download speed (6400 Mbps / 40 Mbps = 160 seconds).Now, convert the download time in seconds to hours, minutes, and seconds: 160 seconds is equal to 2 minutes and 40 seconds.So, the download time is 0 hours, 2 minutes and 40 seconds.

Input Format

The first line of input consists of an integer N, representing the file size in megabytes (MB).

The second line consists of an integer S, representing the network speed in megabits per second(mbps).

Output Format

The output prints "Download Time: X hours, Y minutes, and Z seconds", where X, Y, and Z are integers representing the hours, minutes, and seconds respectively.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 180
3
Output: Download Time: 0 hours, 8 minutes, and 0 seconds

Answer

```
file_size_MB = int(input())
download_speed_Mbps = int(input())

download_time_seconds = (file_size_MB * 8) // download_speed_Mbps
```

```
hours = download_time_seconds // 3600
remaining_seconds = download_time_seconds % 3600
minutes = remaining_seconds // 60
seconds = remaining_seconds % 60

print(f"Download Time: {hours} hours, {minutes} minutes, and {seconds}
seconds")
```

Status : Correct ***Marks : 1/1***

6. Problem Statement

Liam works at a car dealership and is responsible for recording the details of cars that arrive at the showroom. To make his job easier, he wants a program that can take the car's make, model, and price, and display the information in a formatted summary.

Assist him in the program.

Input Format

The first line of input contains a string, representing the car make.

The second line contains a string, representing the car model.

The third line contains a float value, representing the car price.

Output Format

The first line of output prints "Car Make: ", followed by the car make.

The second line prints "Car Model: ", followed by the car model.

The third line prints "Price: ", followed by the car price, formatted to two decimal places.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: Toyota
Camry
23450.75

Output: Car Make: Toyota
Car Model: Camry
Price: Rs.23450.75

Answer

```
car_make = input()
car_model = input()
price = float(input())

print(f"Car Make: {car_make}")
print(f"Car Model: {car_model}")
print(f"Price: Rs.{price:.2f}")
```

Status : Correct ***Marks : 1/1***