Miscellaneous / Combined Questions

Question 7: Basketball Tournament Scoring (Loops and Summation)
Your coach is tracking team performances across several rounds of practice.

a) Counting Rounds (2 marks)
Write a loop that prints the message: “Round X begins!” for each of the 4 rounds.
Expected Output:

```
Round 1 begins!
Round 2 begins!
Round 3 begins!
Round 4 begins!
```

b) Recording and Summing Scores (4 marks)
Modify your program to ask the user for the score in each round (out of 20) and compute the total score.
Example Output:

```
Enter score for Round 1: 15
Enter score for Round 2: 10
Enter score for Round 3: 18
Enter score for Round 4: 12
Total score : 55
```

c) Evaluating Performance (4 marks)
Extend your program to calculate and print:
• The average score per round.
• A message based on the result:
– If the average is 15 or higher → print “Excellent performance”
– Otherwise → print “You need to practice”
Example Output:

```
...( Lines before indicating scores entered for each round ) ...
Total score : 55
Average per round : 13.75
Keep practicing!
```

Question Value: The question familiarizes students with how loops work, how to set up a counter / summation etc.

Question 8: Combining Team Results (Working with Lists)
Two basketball teams played several games. Their scores are stored in sorted lists:
teamA = [45 , 52 , 61 , 70]
teamB = [48 , 55 , 60 , 68 , 73]

a) The Results (5 marks)
The coach wants a summary announcing:
• Each team's total and average score, and
• The "Top Scorer Award" — the single highest score across both teams.
Expected Output:

```
Team A Total : 228 , Average : 57.0
Team B Total : 304 , Average : 60.8
Top Scorer Award : Team B (73 points )
```

b) Merging Sorted Results (4 marks)
Create one combined list that shows all scores in sorted order.
Example Output:

```
[" A - 45" , " B - 48" , " A - 52" , " B - 55" , " B - 60" ,
" A - 61" , " B - 68" , " A - 70" , " B - 73"]
```

Hint: Each team's list is already sorted, you can now just compare the smallest remaining score in each as you merge

Question Value: The first subpart familiarizes students with lists, like list-indexing and iterating over the list. The second subpart is an adaptation of a popular programming problem where we are merging sorted lists, but as a real-world example.

Question 9: Volleyball Court (Nested Lists)
In volleyball, players stand on two sides of the court. We can represent this setup using something called a nested list, which is just a list that contains other lists. Consider the following
Example:

```
court = [
        [47 , 68 , 27 , 79] , # Team A
        [35 , 84 , 21 , 66]   # Team B
]
```

Here the list " courts " is made up of other lists (each list stands for
a team , where each entry in the list is the jersey number for that
player (or player ID)

a) Setting Up the Court (3 marks)
Ask the user how many players are on each team. Then, ask for the player IDs (numbers) for both Team A and Team B, and build the nested list court.

Example Input/Output:

How many players per team ? 4

Enter 4 player IDs for Team A :
47 68 27 79

Enter 4 player IDs for Team B :
35 84 21 66

Court layout :
47 68 27 79
---
35 84 21 66

Hint: Use a list for each team and store each list inside another list to represent the entire court.

b) Accessing Players (3 marks)

Write code that:

• Prints the player ID at position x in Team A's list.

• Prints a range of players from Team B (for example, positions a through c).

Example Input/Output:

Enter position for Team A player : 2
Enter start and end positions for Team B range : 1 3

Team A player at position 2: 27
Team B players from position 1 to 3: [84 , 21 , 66]

Remember: You need to solve this question by assuming you only have your nested list provided. Team A is the first list, and Team B is the second list inside the nested list. Also, Python starts counting from 0, not 1 — so the first position is actually 0, the second is 1, and so on.

c) Switching Sides (4 marks)

Simulate switching sides after a set by swapping the two teams' positions in the court (you could also add a message indicating the switch)

Expected Output:

New court layout :
35 84 21 66
---
47 68 27 79
Teams have switched sides!

The value: Teaches students about nested lists and how to work with them, using a real-life example. The question is broken down into subparts and build up in difficulty.

Question 10: workout_log(reps: list[int]) (Arrays, Loops, Basic Stats)

Track your workout hours by making a list of integers. Each number represents the number of push ups you did on any given day.

Write a function workout_log(reps: list[int]) that prints:

- The total number of push ups done in a week.
- The average number of push ups in a day (as a float).
- The best (maximum) push up count that day.
- A message based on the average:
    - If average ≥ 40: "Great Work!"
    - If 20 ≤ average < 40: "Solid effort."
    - If average < 20: "Keep going!"

Example Output:

```
>>> workout_log([30, 40, 20, 50, 45, 0, 35])
Total push-ups: 220
Average per day: 31.428571428571427
Best day: 50
Solid effort.
```

Question 11: clothing_budget(items: list[str], prices: list[float], balance: float) (Arrays, Conditionals)

You are planning what to buy for the month. You have two lists:

```
items  = ["Shirt", "Sweatpants", "Pair of socks", "Hoodie"]
prices = [12.50, 18.00, 5.25, 22.75]
```

Write a function clothing_budget(selected_indices: list[int], balance: float) that:

- Takes a list of indices and a starter balance as inputs.
- Prints each "item added to cart" when added to cart and keeps track of total.
- After processing all items, prints:
    - The total cost.
    - The balance remaining (balance − total cost).
    - If total cost is greater than the balance, print:
        - "Over budget by $X."
        - Otherwise print:

        - "Order placed!"

Example Output:

```
>>> clothing_budget([0, 2, 2], 30.00)
T-shirt ($12.5) added to cart!
Socks ($5.25) added to cart!
Socks ($5.25) added to cart!
Total cost: $23.0
Order placed!
Current Balance: $7.0
```