Basic Loop Questions

1. Write a program to **print numbers from 1 to 10** using a for loop.
2. Write a program to **print numbers from 10 to 1** using a while loop.
3. Write a program to **print even numbers between 1 and 20** using a loop.
4. Write a program to **print odd numbers between 1 and 20** using a loop.
5. Write a program to **calculate the sum of the first 10 natural numbers** using a loop.
6. Write a program to **calculate the factorial of a number** using a for loop.
7. Write a program to **print the multiplication table of a given number** using a for loop.
8. Write a program to **print numbers from 1 to n** (user input) using a loop.
9. Write a program to **reverse a given number** using a while loop.
10. Write a program to **find the sum of digits of a number** using a loop.

Pattern Printing Questions

11. Print the following pattern:

```
*
**
***
****
*****
```

12. Print the following pattern:

```
*****
****
***
**
*
```

13. Print the following pattern:

```
1
12
123
1234
12345
```

14. Print the following pattern:

```
1
22
333
4444
55555
```

15. Print a pyramid pattern of stars using nested loops.
16. Print a diamond pattern using nested loops.
17. Print a Floyd's Triangle.
18. Print a Pascal's Triangle.
19. Print the following pattern:

```
A
BB
CCC
DDDD
EEEEE
```

20. Print the following pattern:

```
ABCDE
ABCD
ABC
AB
A
```

Loop Control Statements (break, continue, pass)

21. Write a program to **print numbers from 1 to 10, but stop if the number is 7** (using `break`).
22. Write a program to **print numbers from 1 to 10, but skip printing 5** (using `continue`).
23. Write a program to **check if a number is prime** using a for loop.
24. Write a program to **find the first 5 multiples of 3** using a loop.
25. Write a program to **count the number of digits in a number** using a while loop.
26. Write a program to **print the first 10 Fibonacci numbers** using a loop.
27. Write a program to **check if a given string is a palindrome** using a loop.
28. Write a program to **reverse a string using a loop**.
29. Write a program to **find the largest digit in a given number** using a loop.
30. Write a program to **find the smallest digit in a given number** using a loop.

Advanced Loop Questions

31. Write a program to **generate prime numbers from 1 to n** using a loop.
32. Write a program to **find the greatest common divisor (GCD) of two numbers** using a loop.
33. Write a program to **find the least common multiple (LCM) of two numbers** using a loop.
34. Write a program to **print Armstrong numbers from 1 to 1000**.
35. Write a program to **check if a number is perfect (sum of divisors equals the number itself)**.
36. Write a program to **print all factors of a number** using a loop.

37. Write a program to **print numbers from 1 to 100, replacing multiples of 3 with "Fizz", multiples of 5 with "Buzz", and multiples of both with "FizzBuzz"**.
38. Write a program to **find the sum of even and odd numbers separately in a given range**.
39. Write a program to **print the binary equivalent of a decimal number** using a loop.
40. Write a program to **convert a binary number to decimal using a loop**.

Nested Loop Questions

41. Write a program to **print the multiplication table for numbers from 1 to 10** using nested loops.
42. Write a program to **print all possible pairs of numbers from 1 to 5** using a nested loop.
43. Write a program to **find the sum of each row in a given matrix using nested loops**.
44. Write a program to **print prime numbers in a given range using nested loops**.
45. Write a program to **print an inverted right-angled triangle pattern using a nested loop**.
46. Write a program to **find all Pythagorean triplets (a, b, c) where $a^2 + b^2 = c^2$ for numbers up to 50**.
47. Write a program to **sort a list of numbers using the Bubble Sort algorithm** using nested loops.
48. Write a program to **generate a Pascal's triangle using nested loops**.
49. Write a program to **print the ASCII values of uppercase letters using loops**.
50. Write a program to **find the frequency of each digit in a given number using loops**.

Looping with Lists & Strings

51. Write a program to **find the sum of all elements in a list using a loop**.
52. Write a program to **find the maximum and minimum elements in a list using loops**.
53. Write a program to **count the occurrences of a specific element in a list using a loop**.
54. Write a program to **reverse a list using a loop**.
55. Write a program to **find the second largest element in a list using loops**.
56. Write a program to **remove duplicates from a list using a loop**.
57. Write a program to **find the longest word in a given sentence using loops**.
58. Write a program to **find the number of vowels and consonants in a given string**.
59. Write a program to **count the number of words in a given sentence using a loop**.
60. Write a program to **find whether a string is an anagram of another using loops**.

🛠️ Real-World Problem Statement: Smart Parking System Management

Scenario:

A **shopping mall** has a **multi-level parking system** with multiple rows of parking slots on each floor. The mall management wants to implement a **Python-based parking system** that can:

- **Track available and occupied parking slots**
- **Assign parking slots dynamically** when a vehicle arrives
- **Allow vehicles to exit and free up slots**
- **Display the parking status in a structured way**

To achieve this, you need to build a program that uses **nested loops** and **control flow statements** (`if-else` , `break` , `continue`) to manage the parking system efficiently.

🚙 Problem Requirements:

1. **Parking Lot Structure:**
    - The parking lot has **multiple floors**, each with **multiple rows of slots**.
    - Each parking slot can either be **occupied (1)** or **available (0)**.
    - The system should represent the parking lot as a **nested list (2D array)** where:
        - **Rows → Represent parking rows on each floor**
        - **Columns → Represent individual parking slots in each row**
2. **Parking Slot Allocation (Vehicle Entry):**
    - When a vehicle arrives, the system should **search for the first available slot**.
    - If an empty slot is found, the vehicle is assigned to it, and the slot is marked as **occupied (1)**.
    - If no slots are available on a floor, the system checks the **next floor**.
    - If the entire parking lot is full, the system should display `"Parking Full"` .
3. **Vehicle Exit (Slot Release):**
    - When a vehicle exits, the user enters the **floor number** and **slot number**.
    - The system marks that slot as **available (0)**.
    - If an invalid slot number is entered, it should display `"Invalid slot selection"` .
4. **Parking Status Display:**
    - The system should print the **current parking status** of all floors.
    - For better visualization, **nested loops** should iterate over each floor and row, printing the occupancy status (`0` for available, `1` for occupied).
5. **Exit the System:**
    - The program runs in a loop until the user chooses to exit.

🔑 Expected Functionalities Using Nested Loops & Control Flow:

- ✅ **Outer loop:** Iterate over floors
- ✅ **Inner loop:** Iterate over rows and slots
- ✅ **Conditionals (`if-else`)** to check for availability
- ✅ `break` **statement** to stop searching once a slot is found
- ✅ `continue` **statement** to skip unnecessary checks