Mock Interview questions on For loop and While Loop

While Loops:

1. **What is the primary use of a "while" loop in Python?**
   Answer: A "while" loop in Python is used for repeated execution of a block of code as long as a specified condition is true.

2. **Write a "while" loop that counts from 1 to 5 and prints the numbers.**
   Answer:

```
num = 1
while num <= 5:
    print(num)
    num += 1
```

3. **What is an infinite loop, and how can you avoid it when using "while" loops?**
   Answer: An infinite loop is a loop that runs indefinitely because its termination condition is never met. To avoid it, ensure that the condition in a "while" loop will eventually become false.
4. **How do you exit a "while" loop prematurely using a "break" statement?**
   Answer: You can exit a "while" loop prematurely using the "break" statement. It immediately exits the loop and continues with the next statement after the loop.

For Loops:

1. **When is a "for" loop typically used in Python?**

   Answer: A "for" loop is typically used in Python for iterating over a sequence (such as a list, tuple, or string) or for repeating a block of code a specific number of times.

2. **Write a Python "for" loop to iterate over a list of names and print each name.**

```
names = ["Alice", "Bob", "Charlie"]
for name in names:
```

```
print(name)
```

3. Explain the purpose of the "range()" function in "for" loops.

Answer: The "range()" function generates a sequence of numbers that can be used in "for" loops. For example, `range(5)` generates `[0, 1, 2, 3, 4]`.

4. How can you iterate over the keys of a dictionary using a "for" loop?

Answer: To iterate over the keys of a dictionary using a "for" loop, you can use the `.keys()` method or simply loop directly through the dictionary.

5. What is the main difference between "for" loops and "while" loops in Python?

Answer: The main difference between "for" loops and "while" loops is that "for" loops are typically used when you know the number of iterations in advance and want to iterate over a sequence, while "while" loops are used when you want to repeat a block of code as long as a condition is true.

25 Python Loop Coding Questions along with Explanations for each. Let's get started ↓

1. **Print numbers from 1 to 10 using a for loop:**

```
for num in range(1, 11):
    print(num)
```

Explanation: The for loop iterates over the range(1, 11) which generates numbers from 1 to 10 (inclusive) and prints each number.

1. **Calculate the sum of numbers from 1 to 10 using a for loop:**

```
sum_numbers = 0
for num in range(1, 11):
    sum_numbers += num
print(sum_numbers)
```

Explanation: The for loop iterates over the range(1, 11) and adds each number to the sum_numbers variable.

1. **Print the elements of a list using a for loop:**

```
my_list = [1, 2, 3, 4, 5]
for element in my_list:
    print(element)
```

Explanation: The for loop iterates over each element in the my_list and prints them one by one.

1. **Calculate the product of elements in a list using a for loop:**

```
my_list = [2, 3, 4, 5]
product = 1
for num in my_list:
    product *= num
print(product)
```

Explanation: The for loop iterates over each element in the my_list and multiplies them together, updating the product variable.

1. **Print even numbers from 1 to 10 using a for loop:**

```
for num in range(2, 11, 2):
    print(num)
```

Explanation: The for loop iterates over the range(2, 11, 2) which generates even numbers from 2 to 10 (inclusive) and prints each even number.

1. **Print numbers in reverse from 10 to 1 using a for loop:**

```
for num in range(10, 0, -1):
    print(num)
```

Explanation: The for loop iterates over the range(10, 0, -1) which generates numbers in reverse from 10 to 1 (inclusive) and prints each number.

1. **Print characters of a string using a for loop:**

```
my_string = "Hello"
for char in my_string:
    print(char)
```

Explanation: The for loop iterates over each character in the my_string and prints them one by one.

1. **Find the largest number in a list using a for loop:**

```
my_list = [3, 9, 1, 6, 2, 8]
largest = my_list[0]
for num in my_list:
    if num > largest:
        largest = num
print(largest)
```

Explanation: The for loop iterates over each element in the my_list, compares it with the current value of largest, and updates largest if the current element is larger.

1. **Find the average of numbers in a list using a for loop:**

```
my_list = [4, 7, 9, 2, 5]
total = 0
for num in my_list:
    total += num
average = total / len(my_list)
print(average)
```

Explanation: The for loop iterates over each element in the my_list and calculates the sum of all elements. The average is then calculated by dividing the sum by the number of elements in the list.

1. **Print all uppercase letters and display the number of uppercase letters in a string using a for loop:**

```
my_string = "Hello World"
count=0
for char in my_string:
    if char.isupper():
        count=count+1
        print(char)
print("Number of Upper case letters: ", count)
```

Explanation: The for loop iterates over each character in the my_string and checks if it is uppercase using the isupper() method. If it's uppercase, it is printed.

1. **Count the number of vowels in a string using a for loop:**

```
my_string = "Hello World"
vowels = "AEIOUaeiou"
count = 0
for char in my_string:
    if char in vowels:
```

```
        count += 1
print(count)
```

Explanation: The for loop iterates over each character in the my_string and checks if it is a vowel by comparing it with the characters in the vowels string.

1. **Print a pattern of stars using nested for loops:**

```
for i in range(5):
    for j in range(i + 1):
        print("*", end="")
    print()
```

Explanation: The outer for loop iterates from 0 to 4, and the inner for loop prints stars in increasing order for each iteration of the outer loop.

1. **Find the first occurrence of a number in a list using a while loop:**

```
my_list = [3, 8, 2, 7, 4]
target = 7
index = 0
while index < len(my_list):
    if my_list[index] == target:
        break
    index += 1
```

```
else:
    index = -1
print(index)
```

Explanation: The while loop iterates through the my_list and checks if each element is equal to the target. If found, it breaks out of the loop, otherwise, it continues. The else block is executed if the loop completes without finding the target.

1. **Calculate the sum of numbers from 1 to 100 using a while loop:**

```
num = 1
sum_numbers = 0
while num <= 100:
    sum_numbers += num
    num += 1
print(sum_numbers)
```

Explanation: The while loop iterates from 1 to 100, adding each number to the sum_numbers variable.

1. **Find all prime numbers between 1 and 50 using nested for and if:**

```
for num in range(2, 51):
    for i in range(2, num):
        if num % i == 0:
```

```
            break
        else:
            print(num)
```

Explanation: The outer for loop iterates from 2 to 50, and the inner for loop checks for divisibility of num by numbers from 2 to num - 1. If it is divisible by any number, the inner loop is broken, otherwise, the else block is executed, and the number is printed.

1. **Print numbers divisible by 3 or 5 from 1 to 20 using a for loop:**

```
for num in range(1, 21):
    if num % 3 == 0 or num % 5 == 0:
        print(num)
```

Explanation: The for loop iterates from 1 to 20 and checks if each number is divisible by 3 or 5. If true, it prints the number.

1. **Print the Fibonacci sequence up to the 10th term using a while loop:**

```
a, b = 0, 1
count = 0
```

```
while count < 10:
    print(a, end=" ")
    a, b = b, a + b
    count += 1
```

Explanation: The while loop calculates and prints the Fibonacci sequence by updating a and b variables in each iteration.

1. **Find the common elements in two lists using a for loop:**

```
list1 = [1, 2, 3, 4, 5]
list2 = [3, 4, 5, 6, 7]
common_elements = []
for element in list1:
    if element in list2:
        common_elements.append(element)
print(common_elements)
```

Explanation: The for loop iterates over elements in list1 and checks if they are also present in list2. If so, it adds them to the common_elements list.

1. **Print numbers in a list until a negative number is encountered using a while loop:**

```
my_list = [1, 4, 6, 8, 10, -3, 5, 7]
index = 0
```

```
while my_list[index] >= 0:
    print(my_list[index])
    index += 1
```

Explanation: The while loop iterates through the my_list until a negative number is encountered, printing each non-negative number.

1. **Print numbers from 1 to 5, except 3 using a for loop and continue statement:**

```
for num in range(1, 6):
    if num == 3:
        continue
    print(num)
```

Explanation: The for loop iterates from 1 to 5 and skips the number 3 using the continue statement.

1. **Print numbers from 1 to 10. If a number is divisible by 4, stop the loop using a for loop and break statement:**

```
for num in range(1, 11):
    print(num)
    if num % 4 == 0:
        break
```

Explanation: The for loop iterates from 1 to 10 and prints each number. If the number is divisible by 4, the loop is terminated using the break statement.

1. **Print numbers from 1 to 10. If a number is even, skip it using a for loop and else clause:**

```
for num in range(1, 11):
    if num % 2 == 0:
        continue
    print(num)
else:
    print("Loop completed successfully!")
```

Explanation: The for loop iterates from 1 to 10 and skips even numbers using the continue statement. The else clause is executed after the loop completes successfully.

1. **Print numbers from 1 to 10. If a number is even, break the loop using a for loop and else clause:**

```
for num in range(1, 11):
    if num % 2 == 0:
        break
    print(num)
else:
```

```
print("Loop completed successfully!")
```

Explanation: The for loop iterates from 1 to 10 and breaks the loop if it encounters an even number.

1. **Write a program to print the below pattern**

   P

   P Y

   P Y T

   P Y T H

   P Y T H O

   P Y T H O N

Write a program to print the below pattern

```
        1
      1 2
    1 2 3
  1 2 3 4
1 2 3 4 5
```

Write a program to print below pattern