Theory Questions of Python:

1. What is Python, and what are its advantages?
2. What are the basic data types in Python?
3. What is the difference between a list and a tuple in Python?
4. How do you declare a variable in Python?
5. What is the difference between Python 2 and Python 3?
6. What is a dictionary in Python, and how is it different from a list?
7. How do you handle exceptions in Python?
8. What is the difference between a function and a method in Python?
9. What is a lambda function in Python?
10. How do you read and write files in Python?
11. What are decorators in Python?
12. How do you create a virtual environment in Python?
13. What is the difference between a module and a package in Python?
14. How do you perform unit testing in Python?
15. What are the main features of object-oriented programming in Python?
16. What is a loop in Python, and what is its purpose?
17. What are the types of loops in Python?
18. What is the difference between a for loop and a while loop in Python?
19. How do you iterate over a list in Python?
20. How do you use the range function in Python?
21. What is the purpose of the break statement in Python?
22. What is the purpose of the continue statement in Python?
23. How do you iterate over a dictionary in Python?
24. How do you use the enumerate function in Python?
25. How do you use nested loops in Python?
26. How do you use the zip function in Python?
27. How do you use the sorted function in Python?
28. How do you use the reversed function in Python?

29. What is the purpose of the else statement in a loop in Python?
30. What is the purpose of a loop in programming?
31. What is an infinite loop, and how can you avoid it?
32. What is the difference between a for loop and a while loop in Python?
33. What is the purpose of the break keyword in a loop in Python?
34. What is the purpose of the continue keyword in a loop in Python?
35. How do you iterate over a range of numbers in Python?
36. What is the enumerate function, and how is it used in a loop?
37. What is the zip function, and how is it used in a loop?
38. How do you use a for loop to iterate over a string in Python?
39. What is the purpose of the range function in Python, and how is it used in a loop?
40. How do you use nested loops in Python?
41. How do you iterate over a list in reverse order using a loop in Python?
42. What is the purpose of the else statement in a loop in Python?
43. How do you use the sorted function to sort a list in Python?
44. What is the difference between a break statement and a continue statement in Python?
45. How do you use the reversed function to iterate over a sequence in reverse order in Python?
46. What is the purpose of the pass keyword in a loop in Python?
47. How do you use a while loop to iterate over a list in Python?
48. What is an example of a use case for a nested loop in Python?
49. How do you use the range function to iterate over a list in Python, and what are its arguments?

PYTHON PROGRAM FOR PRACTICE: List Questions

1. Write a Python program to create a list of numbers and print it.
2. Write a Python program to find the length of a list.
3. Write a Python program to sum all the elements in a list.
4. Write a Python program to find the largest and smallest element in a list.
5. Write a Python program to find the second largest element in a list.
6. Write a Python program to find the common elements between two lists.
7. Write a Python program to remove duplicates from a list.
8. Write a Python program to reverse a list.
9. Write a Python program to sort a list in ascending order.
10. Write a Python program to sort a list in descending order.
11. Write a Python program to count the occurrences of an element in a list.
12. Write a Python program to insert an element at a specific position in a list.
13. Write a Python program to remove an element from a list.
14. Write a Python program to create a new list containing the first and last elements of a given list.
15. Write a Python program to check if a list is empty or not.
16. Write a Python program to find the intersection of two lists.
17. Write a Python program to find the union of two lists.
18. Write a Python program to find the difference between two lists.
19. Write a Python program to find the frequency of each element in a list.
20. Write a Python program to remove all the elements from a list.
21. Write a Python program to shuffle a list.
22. Write a Python program to create a list of all odd numbers between 1 and 10.
23. Write a Python program to create a list of squares of numbers from 1 to 5.
24. Write a Python program to create a list of even numbers from a given list of integers.
25. Write a Python program to find the second smallest number in a list.
26. Write a Python program to find the sum of all elements in a list using recursion.
27. Write a Python program to reverse a list.

28. Write a Python program to find the maximum and minimum numbers in a list.
29. Write a Python program to sort a list of strings by their length.
30. Write a Python program to find the largest number in a list using recursion.

PYTHON PROGRAM FOR PRACTICE: Loop

50. Print all the numbers from 1 to 10 using a while loop.
51. Print all the even numbers from 1 to 20 using a for loop.
52. Calculate the sum of all the numbers from 1 to 100 using a while loop.
53. Print the first 10 numbers in the Fibonacci sequence using a for loop.
54. Print the multiplication table of 7 using a for loop.
55. Calculate the factorial of a number using a while loop.
56. Print all the prime numbers between 1 and 100 using a for loop.
57. Print the ASCII values of all the characters in a string using a for loop.
58. Reverse a given string using a while loop.
59. Print the squares of the first 10 natural numbers using a for loop.
60. Print the sum of all the odd numbers from 1 to 50 using a while loop.
61. Print the sum of all the numbers in a list using a for loop.
62. Print the first 20 numbers in the sequence 1, 4, 7, 10, 13... using a while loop.
63. Print all the uppercase letters in a string using a for loop.
64. Generate a multiplication table for all the numbers from 1 to 10 using nested for loops.
65. Print the cube of the first 15 even numbers using a for loop.
66. Print the sum of the digits of a number using a while loop.
67. Print the product of all the numbers in a list using a for loop.
68. Print the binary equivalent of a given number using a while loop.
69. Print the sum of the numbers in the range 10 to 20 that are divisible by 3 using a for loop.
70. Print the first 10 even numbers.
71. Print the first 10 odd numbers.
72. Calculate the sum of the first 100 natural numbers.

73. Print the first 10 numbers in the sequence 1, 3, 5, 7, 9...
74. Print the first 20 numbers in the sequence 1, 1, 2, 3, 5, 8...
75. Calculate the factorial of a given number.
76. Print all the prime numbers between 1 and 100.
77. Print the ASCII values of all the characters in a string.
78. Reverse a given string.
79. Print the squares of the first 10 natural numbers.
80. Print the sum of all the even numbers between 1 and 100.
81. Print the sum of all the odd numbers between 1 and 50.
82. Print the sum of all the numbers in a list.
83. Print the first 20 numbers in the sequence 1, 4, 7, 10, 13...
84. Generate a multiplication table for all the numbers from 1 to 10 using nested loops.
85. Print the cube of the first 15 even numbers.
86. Print the sum of the digits of a given number.
87. Print the product of all the numbers in a list.
88. Print the binary equivalent of a given number.
89. Print the sum of the numbers in the range 10 to 20 that are divisible by 3

Program Based on Function:

1. Write a Python function to find the maximum of three numbers.
2. Write a Python function to sum all the numbers in a list.
3. Write a Python function to multiply all the numbers in a list.
4. Write a Python program to reverse a string.
5. Write a Python function to check whether a number falls within a given range.
6. Write a Python function to calculate the factorial of a number (a non-negative integer). The function accepts the number as an argument.
7. Write a Python function that accepts a string and counts the number of upper and lower case letters.

8. Write a Python function that takes a list and returns a new list with distinct elements from
9. Write a Python function that takes a number as a parameter and checks whether the number is prime or not.
10. Note: A prime number (or a prime) is a natural number greater than 1 and that has no positive divisors other than 1 and itself.
11. Write a Python function that prints out the first n rows of Pascal's triangle.
12. Write a Python program that accepts a hyphen-separated sequence of words as input and prints the words in a hyphen-separated sequence after sorting them alphabetically.
13. Write a Python function to create and print a list where the values are the squares of numbers between 1 and 30 (both included).

OOPS Practice:

Task 1

create a class with methods to

1. check if a number is pallindrome
2. find the factorial value
3. generate a fibonacci series of len of the number
4. generate a series of prime numbers upto the number
5. check if a number is armstrong number

Task 2

create a class with methods to

1. count the element of a list
2. concatenate the list elements as strings
3. reverse a list
4. find the second highest in list
5. sort the list of numbers
6. sort the list of alphabets

Task 3

create a abstract class with a abstract method to find the palindrome

1. Every school has a basic structure with below mentioned methods
    - admission- show how many admissions done
    - teaching- how many teachers are allotted
    - examination- how many students successfully completed the examination
    - results- total students passed and failed

making use of the above structure show the details of below school

1. Podar school - 10 percent students do not attend examination, 5 percent failed of the students attempted
2. MG public school- 8 percent students do not attend examination, 6 percent failed of the students attempted
3. Vidyaniketan School- 12 percent students do not attend examination, 15 percent failed of the students attempted
4. Hindi Bal Vidya Mandir- 5 percent students not attend examination, 2 percent failed of the students attempted

Explanation of OOPS:

OOPS stands for Object-Oriented Programming System, which is a programming paradigm that revolves around the concept of objects, rather than functions or procedures. In OOPS, data and functions are organized into objects that can interact with one another to perform tasks or solve problems.

The main principles of OOPS are:

1. **Abstraction**: Abstraction is the process of representing complex real-world objects in a simplified manner by hiding irrelevant details. It allows the programmer to focus on the essential features of an object and ignore the rest.
2. **Encapsulation:** Encapsulation is the process of bundling data and functions together into a single unit, known as a class. This allows the data to be protected from unwanted access and modification.
3. **Inheritance:** Inheritance is the process of creating a new class from an existing one, known as the parent class or base class. The new class, known as the child class or derived class, inherits the properties and methods of the parent class.
4. **Polymorphism:** Polymorphism is the ability of an object to take on many forms. It allows objects of different classes to be treated as if they were of the same class.

OOPS provides several advantages over traditional procedural programming, including:

- Reusability: OOPS allows code to be reused by creating objects that can be used in multiple programs.
- Modularity: OOPS allows the code to be divided into smaller, more manageable parts.
- Flexibility: OOPS allows for easy modification and extension of existing code.
- Better organization: OOPS provides a better way to organize code and data, making it easier to maintain and debug.