1. Write a Python program to reverse a string without using any built-in string reversal functions.

Ans. def reverse string(string):

return string[::-1]

2. Implement a function to check if a given string is a palindrome.

Ans. def isPalindrome(str):

for i in range(0, int(len(str)/2)):

if str[i] != str[len(str)-i-1]:

return False

return True

s = "malayalam"

ans = isPalindrome(s)

if (ans):

print("Yes")

else:

print("No")

3. Write a program to find the largest element in a given list.

Ans. list1 = [10, 20, 4, 45, 99]

print("Largest element is:", max(list1))

4. Implement a function to count the occurrence of each element in a list.

Ans. l = [8, 6, 8, 10, 8, 20, 10, 8, 8]

l.count(8)

5. Write a Python program to find the second largest number in a list.

Ans. l1 = [10, 20, 20, 4, 45, 45, 45, 99, 99]

l2=list(set(l1))

l2.sort()

print("second largest element is:-", l2[-2])

6. Implement a function to remove duplicate elements from a list.

Ans. l = [1, 2, 4, 2, 1, 4, 5]

print("Original List: ", l)

res = [\*set(l)]

print("after removing the duplicates:- ", res)

7. Write a program to calculate the factorial of a given number.

Ans. for i in range(1,num + 1):

    factorial = factorial\*i

print("The factorial of",num,"is",factorial)

8. Implement a function to check if a given number is prime.

Ans. if num == 1:

    print(num, "is not a prime number")

elif num > 1:

    # check for factors

    for i in range(2, num):

        if (num % i) == 0:

            break

    else:

        print(num, "is a prime number")

9. Write a Python program to sort a list of integers in ascending order.

Ans. l = [1, 2, 4, 2, 1, 4, 5]

l.sort()

print("sorted list: ", l)

10. Implement a function to find the sum of all numbers in a list.

Ans. l = [1, 2, 4, 2, 1, 4, 5]

sum = sum(l)

print("sum of the all elements in list:- ", sum)

11. Write a program to find the common elements between two lists.

Ans. def common\_member(a, b):

    a\_set = set(a)

    b\_set = set(b)

    if (a\_set & b\_set):

        print(a\_set & b\_set)

    else:

        print("No common elements")

a = [1, 2, 3, 4, 5]

b = [5, 6, 7, 8, 9]

common\_member(a, b)

a = [1, 2, 3, 4, 5]

b = [6, 7, 8, 9]

common\_member(a, b)

12. Implement a function to check if a given string is an anagram of another string.

Ans. class Solution:

    # Function is to check whether two strings are anagram of each other or not.

    def isAnagram(self, a, b):

        if sorted(a) == sorted(b):

            return True

        else:

            return False

if \_\_name\_\_ == '\_\_main\_\_':

    a = "gram"

    b = "arm"

    if(Solution().isAnagram(a, b)):

      print("The two strings are anagram of each other")

    else:

      print("The two strings are not anagram of each other")

13. Write a Python program to generate all permutations of a given string.  
Ans. def permute(s, answer):

    if (len(s) == 0):

        print(answer, end = "  ")

        return

    for i in range(len(s)):

        ch = s[i]

        left\_substr = s[0:i]

        right\_substr = s[i + 1:]

        rest = left\_substr + right\_substr

        permute(rest, answer + ch)

# Driver Code

answer = ""

s = input("Enter the string : ")

print("All possible strings are : ")

permute(s, answer)

14. Implement a function to calculate the Fibonacci sequence up to a given number of terms.

Ans. def fibonacci(n):

    a = 0

    b = 1

    if n < 0:

        print("Incorrect input")

    elif n == 0:

        return a

    elif n == 1:

        return b

    else:

        for i in range(2, n+1):

            c = a + b

            a = b

            b = c

        return b

15. Write a program to find the median of a list of numbers.

Ans. import statistics

# initializing list

test\_list = [4, 5, 8, 9, 10, 17]

# printing list

print("The original list : " + str(test\_list))

# Median of list

# Using statistics.median()

res = statistics.median(test\_list)

# Printing result

print("Median of list is : " + str(res))

16. Implement a function to check if a given list is sorted in non-decreasing order.  
Ans. def arraySortedOrNot(arr):

    # Calculating length

    n = len(arr)

    if n == 1 or n == 0:

        return True

    return arr[0] <= arr[1] and arraySortedOrNot(arr[1:])

arr = [20, 23, 23, 45, 78, 88]

# Displaying result

if arraySortedOrNot(arr):

    print("Yes")

else:

    print("No")

17. Write a Python program to find the intersection of two lists.

Ans. def intersection(lst1, lst2):

    lst3 = [value for value in lst1 if value in lst2]

    return lst3

# Driver Code

lst1 = [4, 9, 1, 17, 11, 26, 28, 54, 69]

lst2 = [9, 9, 74, 21, 45, 11, 63, 28, 26]

print(intersection(lst1, lst2))

18. Implement a function to find the maximum subarray sum in a given list.

Ans. import sys

from sys import maxint

def maxSubArraySum(a, size):

    max\_so\_far = -maxint - 1

    max\_ending\_here = 0

    for i in range(0, size):

        max\_ending\_here = max\_ending\_here + a[i]

        if (max\_so\_far < max\_ending\_here):

            max\_so\_far = max\_ending\_here

        if max\_ending\_here < 0:

            max\_ending\_here = 0

    return max\_so\_far

a = [-2, -3, 4, -1, -2, 1, 5, -3]

print "Maximum contiguous sum is", maxSubArraySum(a, len(a))

19. Write a program to remove all vowels from a given string.

Ans. def rem\_vowel(string):

    vowels = ['a','e','i','o','u']

    result = [letter for letter in string if letter.lower() not in vowels]

    result = ''.join(result)

    print(result)

# Driver program

string = "my name is manish and i am solving python assignment"

rem\_vowel(string)

20. Implement a function to reverse the order of words in a given sentence.

Ans. s = "my name is manish and i am solving python assignments"

words = s.split(' ')

string = []

for word in words:

    string.insert(0, word)

print(" ".join(string))

21. Write a Python program to check if two strings are anagrams of each other.

Ans. def check(s1, s2):

    # the sorted strings are checked

    if(sorted(s1)== sorted(s2)):

        print("The strings are anagrams.")

    else:

        print("The strings aren't anagrams.")

# driver code

s1 ="bad"

s2 ="dad"

check(s1, s2)

22. Implement a function to find the first non-repeating character in a string.

Ans. def FirstNonRepeat(s):

    for i in s:

        if (s.find(i, (s.find(i)+1))) == -1:

            print("First non-repeating character is", i)

            break

    return

# \_\_main\_\_

s = 'inueron'

FirstNonRepeat(s)

23. Write a program to find the prime factors of a given number.

Ans.

24. Implement a function to check if a given number is a power of two.

Ans. def powerof2(n):

  if n == 1:

        return True

  elif n % 2 != 0 or n == 0:

        return False

  return powerof2(n/2)

if \_\_name\_\_ == "\_\_main\_\_":

        # Function call

    print(powerof2(64))

    print(powerof2(12))

25. Write a Python program to merge two sorted lists into a single sorted list.

Ans. test\_list1 = [1, 5, 6, 9, 11]

test\_list2 = [3, 4, 7, 8, 10]

# printing original lists

print ("The original list 1 is : " + str(test\_list1))

print ("The original list 2 is : " + str(test\_list2))

# using sorted()

# to combine two sorted lists

res = sorted(test\_list1 + test\_list2)

# printing result

print ("The combined sorted list is : " + str(res))

26. Implement a function to find the mode of a list of numbers.

Ans. number\_list = [1, 2, 2, 3, 4, 4, 5, 5, 6, 7, 8, 8, 8]

uniq\_values = []

mode\_values = []

for i in number\_list:

  if i not in uniq\_values:

    uniq\_values.append(i)

  else:

    mode\_values.append(i)

print(set(mode\_values))

27. Write a program to find the greatest common divisor (GCD) of two numbers.

Ans. def gcd(a, b):

  result = min(a, b)

  while result:

    if a % result == 0 and b % result == 0:

      break

    result -= 1

  return result

if \_\_name\_\_ == '\_\_main\_\_':

  a = 98

  b = 56

  print(f"GCD of {a} and {b} is {gcd(a, b)}")

28. Implement a function to calculate the square root of a given number.

Ans. import math

print(math.sqrt(9))

29. Write a Python program to check if a given string is a valid palindrome ignoring non-alphanumeric characters.

Ans. import re

def is\_valid\_palindrome(s):

    # Remove non-alphanumeric characters from the string

    cleaned\_string = re.sub(r'[^a-zA-Z0-9]', '', s)

    # Convert the cleaned string to lowercase

    cleaned\_string = cleaned\_string.lower()

    # Check if the cleaned string is equal to its reverse

    return cleaned\_string == cleaned\_string[::-1]

# Test the function

input\_string = input("Enter a string: ")

if is\_valid\_palindrome(input\_string):

    print("The string is a valid palindrome.")

else:

    print("The string is not a valid palindrome.")

30. Implement a function to find the minimum element in a rotated sorted list.

Ans. def findMin(arr, N):

    min\_ele = arr[0];

    for i in range(N) :

        if arr[i] < min\_ele :

            min\_ele = arr[i]

    return min\_ele;

# Driver program

arr = [5, 6, 1, 2, 3, 4]

N = len(arr)

print(findMin(arr,N))

31. Write a program to find the sum of all even numbers in a list.

Ans. def fun(n):

    sum = 0

    i = 2

    while i <= n:

        sum += i

        i = i + 2

    return sum

n = int(input('Enter any number: '))

print("Sum of all even numbers from 1 to", n , "is: " , fun(n))

32. Implement a function to calculate the power of a number using recursion.

Ans. def power(N, P):

  if P == 0:

        return 1

  return (N\*power(N, P-1))

  if \_\_name\_\_ == '\_\_main\_\_':

    N = 5

    P = 2

    print(power(3, 3))

33. Write a Python program to remove duplicates from a list while preserving the order.

Ans. l = [1, 2, 4, 2, 1, 4, 5]

print("Original List: ", l)

res = [\*set(l)]

print("List after removing duplicate elements: ", res)

34. Implement a function to find the longest common prefix among a list of strings.

Ans. def longestCommonPrefix(my\_str):

    if my\_str == []:

        return ''

    if len(my\_str) == 1:

        return my\_str[0]

    my\_str.sort()

    shortest = my\_str[0]

    prefix = ''

    for i in range(len(shortest)):

        if my\_str[len(my\_str) - 1][i] == shortest[i]:

            prefix += my\_str[len(my\_str) - 1][i]

        else:

            break

    return prefix

my\_list\_1 = ['car', 'carbon', 'vehicle']

my\_list\_2 = ['car', 'carbon', 'carbonmonoxide']

# this will give an emptry string

print(longestCommonPrefix(my\_list\_1))

# this will return the string 'car'

print(longestCommonPrefix(my\_list\_2))

35. Write a program to check if a given number is a perfect square.

Ans. import math

def isPerfectSquare(x):

    #if x >= 0,

    if(x >= 0):

        sr = int(math.sqrt(x))

        # sqrt function returns floating value so we have to convert it into integer

        #return boolean T/F

        return ((sr\*sr) == x)

    return false

# Driver code

x = 2502

if (isPerfectSquare(x)):

    print("Yes")

else:

    print("No")

36. Implement a function to calculate the product of all elements in a list.

Ans. import numpy

def multiply\_numbers(list):

   return numpy.prod(list)

given\_list = [2,5,3,7,4,85,-3]

print('The list is:',given\_list)

print("The product is: ")

print(multiply\_numbers(given\_list))

37. Write a Python program to reverse the order of words in a sentence while preserving the word order.

Ans. def reverseWordSentence(Sentence):

    # All in One line

    return ' '.join(word[::-1] for word in Sentence.split(" "))

# Driver's Code

Sentence = "Geeks for Geeks"

print(reverseWordSentence(Sentence))

38. Implement a function to find the missing number in a given list of consecutive numbers.

Ans. def search\_missing\_item(A):

    n = len(A)

    left, right = 0, n - 1

    mid = 0

    while (right > left):

        mid = left + (right - left) // 2

        if (A[mid] - mid == A[0]):

            if (A[mid + 1] - A[mid] > 1):

                return A[mid] + 1

            else:

                left = mid + 1

        else:

            if (A[mid] - A[mid - 1] > 1):

                return A[mid] - 1

            else:

                right = mid - 1

    return -1

A = [1, 2, 3, 4, 5, 6, 7, 9]

print(search\_missing\_item(A))

39. Write a program to find the sum of digits of a given number.

Ans. def getSum(n):

    sum = 0

    while (n != 0):

        sum = sum + int(n % 10)

        n = int(n/10)

    return sum

# Driver code

if \_\_name\_\_ == "\_\_main\_\_":

    n = 687

    # Function call

    print(getSum(n))

40. Implement a function to check if a given string is a valid palindrome considering case sensitivity.

Ans. def isPalindrome(s):

  return s == s[::-1]

# Driver code

s = "malayalam"

ans = isPalindrome(s)

if ans:

  print("Yes")

else:

  print("No")

41. Write a Python program to find the smallest missing positive integer in a list.  
Ans. def getMEX(a):

    found = False

    n = len(a)

    for i in range(1, n + 2):

        found = False

        for j in range(n):

            if a[j] == i:

                found = True

                break

        if found == False:

            return i

42. Implement a function to find the longest palindrome substring in a given string.

Ans. def longestPalindrome(s: str) -> str:

  length = len(s)

  index = -1

  maxlength = 0

  # looping over the string for substrings

  for i in range(length):

    for j in range(i, length):

      ispalindrome = 1

      # checking if string is a palindrome

      for k in range(0, ((j - i) // 2) + 1):

        if s[i + k] != s[j - k]:

          ispalindrome = 0

      # if the string is palindrome update maximum length

      if ispalindrome != 0 and j - i + 1 > maxlength:

        index = i

        maxlength = j - i + 1

  # return the substring from updated index till length maxlength

  return s[index:index + maxlength]

if \_\_name\_\_ == "\_\_main\_\_":

  word = "abcbedrardea"

  print(longestPalindrome(word))

43. Write a program to find the number of occurrences of a given element in a list.

Ans. def countX(lst, x):

    count = 0

    for ele in lst:

        if (ele == x):

            count = count + 1

    return count

# Driver Code

lst = [8, 6, 8, 10, 8, 20, 10, 8, 8]

x = 8

print('{} has occurred {} times'.format(x, countX(lst, x)))

44. Implement a function to check if a given number is a perfect number.

Ans. num=int(input("Enter the number: "))

sum\_v=0

for i in range(1,num):

    if (num%i==0):

        sum\_v=sum\_v+i

if(sum\_v==num):

    print("The entered number is a perfect number")

else:

    print("The entered number is not a perfect number")

45. Write a Python program to remove all duplicates from a string.

Ans. string="geeksforgeeks"

p=""

for char in string:

  if char not in p:

    p=p+char

print(p)

k=list("geeksforgeeks")

46. Implement a function to find the first missing positive

Ans. def first\_missing\_positive (X, n):

  for i in range(1, n + 2):

    missing\_flag = True

    for j in range(n):

      if X[j] == i:

        missing\_flag

        break

    if missing\_flag:

      return i