1. **STRING OPERATIONS AND METHODS**

1.Write a program to find the number of special characters in the given statement

Sample Input:

Given statement: Modi Birthday @ September 17, #&$% is the wishes code for him.

Sample Output:

Number of special Characters: 5

#Python code to Count Alphabets, Special character Numeric values and space

string=input("Please enter a string: ")#take input from the user

alphabets,num,special,space=0,0,0,0;#variable declaration and initilization

a=[]

d=[]

spl=[]

for i in range(len(string)):

if(string[i].isalpha()): #check Alphabets letters

#print(string[i],end="")

alphabets+=1

a.append(string[i])

elif(string[i].isdigit()):#check numeric value

#print(string[i],end="")

num+=1

d.append(string[i])

elif(string[i].isspace()):#check space

space+=1

else:

#print(string[i],end="")

special+=1

spl.append(string[i])

print("Alpabets letters: ",alphabets, a)

print("\nnumbers: ",num, d)

print("\nSpace: ",space)

print("\nSpecial characters: ",special, spl)

2. Write a program to print the number of vowels and number of consonants in the given statement and which is maximum?

Sample Input:

Saveetha School of Engineering Sample

Output:

Number of vowels = 12 Number of Consonants = 15

str = input("Enter the string:")

vcount, ccount= 0,0

Vowels = "AaEeIiOoUu"

c=[]

v=[]

#Converting entire string to lower case to reduce the comparisons

#str = str.lower()

for i in range(0,len(str)):

#Checks whether a character is a vowel

if str[i] in (Vowels):

vcount = vcount + 1

v.append(str[i])

#count = [each for each in str if each in Vowels]

elif (str[i] !=" " and str[i] not in (Vowels)):

ccount = ccount + 1

c.append(str[i])

print("Total number of vowel and consonant are" );

print(vcount,v)

print(ccount,c)

3. Program to find whether two strings have same character in same index and returns the number of matches

Sample input:

S1=”what”

S2=”watch”

Sample output:

1

def match(s1,s2):

count=0

for i in range(min(len(s1),len(s2))):

if s1[i].lower()==s2[i].lower():

count=count+1

return count

#Driver Program

S1="What"

S2="watch"

print("Total number of matches:")

print(match(S1,S2))

4. Program to print number of words in a line and number of lines in a para

Sample input:

'''This is the most straightforward way to count the number

of lines in a text file in Python. The readlines() method reads all

lines from a file and stores it in a list. Next, use the len() function

to find the length of the list which is nothing but total lines present in a file.'''

Sample output:

Number of lines: 3

Number of words in each line:

Line 1 18

Line 2 15

Line 3 22

#Program to print number of words in a line and number of lines in a para

string='''This is the most straightforward way to count the number

of lines in a text file in Python. The readlines() method reads all

lines from a file and stores it in a list. Next, use the len() function

to find the length of the list which is nothing but total lines present in a file.'''

str1=string.split(".")

str1.pop()

print("Number of lines: ",len(str1))

print("Number of words in each line:")

for i in range(len(str1)):

words=str1[i].split()

#print(words)

print("Line",i+1,len(words))

5. Program to find number of sentences starts with "B"

Sample input:

'''The apple doesn't fall. ...

All that glitters are not gold. ...

A picture is worth a thousand words. ...

Beggers can't be choosers. ...

A bird in the hand. ...

Better safe than sorry. ...

An apple a day keeps doctor away. ...

Blood is thicker than water. ...'''

Sample output:

Total number of lines: 8

Number of Sentences that start with letter B : 3

# Program to find number of sentences starts with "B"

string=input(“Enter the Para: ”)

str1=string.split(" ...")

str1.pop()

print("Total number of lines:",len(str1))

count=0

for i in str1:

str2=i.split()

#print(str2)

for j in str2:

if j[0]=="B":

count=count+1

print("Number of Sentences that start with letter B :",count)

6. Write a program that finds whether a given character is present in a string or not. In case it is present it prints the index at which it is present. Do not use built-in find functions to search the character.

Sample Input:

Enter the string: I am a programmer Enter the character to be searched: p

Sample Output:

P is found in string at index: 8

Note: Check for non-available Character in the given statement as Hidden Test case.

str = input("Enter the String:")

# Character to find

c = input("Enter the character to find:")

# Using Naive Method

res = None

j=0

while j<len(str):

for i in range(0,len(str),1):

if str[i] == c:

res = True

print(str[i], "Index:",i)

j=j+1

if res==None:

print("Character not found")

7. Write a program to arrange the letters of the word alphabetically in Normal order and reverse order

Sample Input:

Enter the word: MOSQUE

Sample Output:

Alphabetical Order Normal: E M O Q S U Alphabetical Order Reverse: U S Q O M E

str=input("Enter the string:")

str=str.upper()

sort\_str=sorted(str)

print(sort\_str)

join\_str="".join(sort\_str)

rev\_str=join\_str[::-1]

print(join\_str)

print(rev\_str)

8. Write a program to find the number of letters repeatedly present in the given word and print the Repeated letters.

Sample Input:

Enter the word: TEMPLE Sample Output:

Number of repeated letters = 1 Repeated letter = E

string = input("Enter the string:")

string=string.lower()

repeat=[]

print("Duplicate characters in a given string: ");

#Counts each character present in the string

for i in range(0, len(string)):

count = 1

for j in range(i+1, len(string)):

if(string[i] == string[j] and string[i] != ' '):

count = count + 1;

#Set string[j] to 0 to avoid printing visited character

string = string[:j] + '0' + string[j+1:]

#A character is considered as duplicate if count is greater than 1

if(count > 1 and string[i] != '0'):

repeat.append(string[i])

print(string[i],count)

print("Number of repeated characters:", len(repeat),repeat)

9. Write functions to perform the following String operations and identify the vowels count in string S3.

Sample input: Index: 1

S1=’welcome’ S2=’Homely’

Sample output: wHeolmceolmye

s1 = "welcome"

s2 = "homely"

n = int(input("n="))

output = ""

i = 0

j = 0

while i < len(s1) and j < len(s2):

output += s1[i:i+n] + s2[j:j+n]

i += n

j += n

output += s1[i:] + s2[j:]

print(output)

10. Write a program that accepts a string from user and re displays the same string after removing vowels from it.

Sample Input & Output:

Enter a string: we can play the game The string without vowels is: w cn ply th gm

Sol:

text = input("Enter the String: ")

vowels = ['a', 'e', 'i', 'o', 'u', 'A', 'E', 'I', 'O', 'U']

newtext = ""

for i in range(len(text)):

if text[i] not in vowels:

newtext = newtext + text[i]

print("\nString after removing Vowels: ")

text = newtext

print(text)

11. Given two strings “s” and “t”, determine if they are isomorphic.

Input: s = "egg", t = "add"

Output: true

def isisomorphic(str1, str2):

if len(str1) != len(str2):

return False

else:

map1, map2 = {}, {}

for i in range(len(str1)):

ch1, ch2 = str1[i], str2[i]

if ch1 not in map1:

map1[ch1] = ch2

print(map1)

if ch2 not in map2:

map2[ch2] = ch1

print(map2)

if map1[ch1] != ch2 or map2[ch2] != ch1:

return False

return True

str1 = input("String 1=")

str2 = input("String 2=")

print(isisomorphic(str1, str2))

12. Given an integer n, return the number of strings of length n that consist only of vowels

(a, e, i, o, u) and are lexicographically sorted.

Input: n = 2

Output: 15

def countstrings(n, start):

if n == 0:

return 1

cnt = 0

for i in range(start, 5):

# decrease the length of string

cnt += countstrings(n - 1, i)

return cnt

def countVowelStrings(n):

return countstrings(n, 0)

n = int(input("n="))

print(countVowelStrings(n))

13. Given a string S consisting of N lowercase alphabets, the task is to modify the string S by

replacing each character with the alphabet whose circular distance from the character is equal

to the frequency of the character in S.

Input: S=“ghee”

Output: higg

def modify\_string(S):

frequency = {}

# Count the frequency of each character

for char in S:

frequency[char] = frequency.get(char, 0) + 1

result = ""

# Replace characters with the corresponding circular distance

for char in S:

circular\_distance = ord(char) + frequency[char]

if circular\_distance > 122:

circular\_distance -= 26

result += chr(circular\_distance)

return result

# Example usage:

S = "ghee"

modified\_string = modify\_string(S)

print(modified\_string) # Output: higg

14. Given two strings S1 and S2, representing sentences, the task is to print both sentences after removing all words which are present in both sentences

Input: S1 = “sky is blue in color”, S2 =”Raj likes sky blue color “

Output: is in

Raj likes

def removeCommonWords(s1,s2):

com=[]

sent1=list(s1.split())

sent2=list(s2.split())

for i in sent1:

if i in sent2:

sent1.remove(i)

sent2.remove(i)

com.append(i)

continue

print(\*sent1)

print(\*sent2)

print("common words",\*com)

sentence1 = input("Enter string1: ")

sentence2 = input("Enter string2: ")

removeCommonWords(sentence1,sentence2)

15. Given a string s consisting of words and spaces, return *the length of the* ***last*** *word in the*

*string.* A **word** is a maximal substring consisting of non-space characters only.

Test Case:

Input: s = "Hello World"

Output: 5

s=input("Enter the string:")

s1=s.split()

n=len(s1)

print("Number of words: ",n)

print("Last word: ",s1[n-1], len(s1[n-1]))

16. Given a string s and an integer k, return the length of the longest substring of s such that the frequency of each character in this substring is greater than or equal to k.

s consists of only lowercase English letters.

Test cases:

1.Input: s = "aaabb", k = 3

Output: 3

def Substring(s):

ans, temp = 1, 1

for i in range(1, len(s)):

if (s[i] == s[i - 1]):

temp += 1

else:

ans = max(ans, temp)

temp = 1

ans = max(ans, temp)

return ans

s = input("Enter the string: ")

print(Substring(s))

17. Reverse Words in a String

Given an input string s, reverse the order of the words.

Input: s = "the sky is blue"

Output: "blue is sky the"

str1=input("Enter the string: ")

str2=str1.split()[::-1]

print(\*str2)

18. Raju, has again started troubling people in your city. The people have turned on to you for

getting rid of Raju. Raju presents to you a number consisting of numbers from 0 to 9

characters. He wants you to reverse it from the final answer such that the number becomes

Mirror number. A Mirror is a number which equals its reverse. The hope of people are on you

so you have to solve the riddle. You have to tell if some number exists which you would

reverse to convert the number into Mirror

Sample input:

Enter the number: 123456

Sample output:

Mirror image: 654321

num= int(input("Enter the integer: "))

num1=str(num)

num2=num1[::-1]

print(num2)

19. Given an array of strings strs, group **the anagrams** together. You can return the answer

in **any order**.

Input: strs = ["eat","tea","tan","ate","nat","bat"]

Output: [["bat"],["nat","tan"],["ate","eat","tea"]]

def Anagrams(li ):

dictionary = {}

for word in li:

sortedWord = ''.join(sorted(word))

print(sortedWord)

if sortedWord not in dictionary:

dictionary[sortedWord] = [word]

else:

dictionary[sortedWord] += [word]

return [dictionary[i] for i in dictionary]

li = ['pop','bat','tab','opp','cat']

print(Anagrams(li))

20. Program to print first letters of the word in a sentence separated by dot.

Sample input: "The cat on the wall"

Sample output: T.C.O.T.W.

string="The cat on the wall"

l1=list(string.split())

print(l1)

for i in range(len(l1)):

print(l1[i][0].upper(),end=".")

continue

21. Valid Palindrome

A phrase is a palindrome if, after converting all uppercase letters into lowercase letters and

removing all non-alphanumeric characters, it reads the same forward and backward.

Alphanumeric characters include letters and numbers.

Given a string s, return true if it is a palindrome, or false otherwise.

**Test Cases:**

1.Input: s = "A man, a plan, a canal: Panama"

Output: true

n="A man, a plan, a canal: Panam"

s=n.lower()

text=''

for i in s:

if i.isalpha():

text+=i

x=text[::-1]

if(x==text):

print("Valid Palindrome")

else:

print("Invalid Palindrome")

22. Write a function delchar(s,c) that takes as input strings s and c, where c has length 1 (i.e., a single character), and returns the string obtained by deleting all occurrences of c in s. If c has a length other than 1, the function should return s.

Sample Input:

Enter the string: Hello world

Enter a character to be deleted: l

Sample output:

String after the character is removed: Heo Word

#Display String after removing the given character

text = input("Enter the String: ")

char= input("Enter the char: ")

newtext = ""

for i in range(len(text)):

if text[i]!=char:

newtext = newtext + text[i]

print("\nString after removing the char: ")

text = newtext

print(text)

23. Given two strings haystack and needle, return the index of needle in haystack, if not return -1.

Sample input:

Haystack=’sadbutsad’

Needle=’sad’

Sample output:

[0,6]

def strStr(haystack,needle):

l=[]

if needle==" ":

return 0

else:

for i in range(len(haystack)):

if haystack[i]==needle[0]:

if haystack[i:i+len(needle)]==needle:

l.append(i)

continue

else:

return -1

return l

# Driver Program

haystack="sadsad"

needle="sad"

print(strStr(haystack,needle))

24. Write a python program to evaluate math expression w/o eval().

def evaluate(string):

string = string.replace(" ", "")

def splitby(string, separators):

lis = []

current = ""

for ch in string:

if ch in separators:

lis.append(current)

lis.append(ch)

current = ""

else:

current += ch

lis.append(current)

return lis

lis = splitby(string, "+-")

def evaluate\_mul\_div(string):

lis = splitby(string, "x/")

if len(lis) == 1:

return lis[0]

output = float(lis[0])

lis = lis[1:]

while len(lis) > 0:

operator = lis[0]

number = float(lis[1])

lis = lis[2:]

if operator == "x":

output \*= number

elif operator == "/":

output /= number

return output

for i in range(len(lis)):

lis[i] = evaluate\_mul\_div(lis[i])

output = float(lis[0])

lis = lis[1:]

while len(lis) > 0:

operator = lis[0]

number = float(lis[1])

lis = lis[2:]

if operator == "+":

output += number

elif operator == "-":

output -= number

return output

# Main Program

testcases = "1+2x3-4"

print(evaluate(testcases))

26. Largest 3 digit Palindrome

# Largest Palindrome

n = 0

for a in range(999, 100, -1):

for b in range(a, 100, -1):

x = a \* b

if x > n:

s = str(a \* b)

if s == s[::-1]:

n = a \* b

print(n)

27. Given string num representing a non-negative integer num, and an integer k, return the smallest possible integer after removing k digits from num.

Input: num = "1432219", k = 3

Output: "1219"

def removeKdigits(num,k):

stack = []

for digit in num:

while k > 0 and len(stack) > 0 and stack[-1] > digit:

k -= 1

stack.pop()

stack.append(digit)

if k > 0:

stack = stack[:-k]

return "".join(stack).lstrip("0") or "0"

num="143219"

k=2

print(removeKdigits(num,k))

28. Return the Unicode of Uppercase letters

import string

import re

alphabets = list(string.ascii\_uppercase)

for i in alphabets:

print(i,"=",ord(i))

print(chr(65))

29. Given two strings s1 and s2, write a function that will convert s1 to s2(if possible) by using min conversion.

def editDistance(str1, str2, m, n):

if m == 0:

return n

if n == 0:

return m

if str1[m-1] == str2[n-1]:

return editDistance(str1, str2, m-1, n-1)

return 1 + min(editDistance(str1, str2, m, n-1), # Insert

editDistance(str1, str2, m-1, n), # Remove

editDistance(str1, str2, m-1, n-1) # Replace

)

# Driver code

str1 = "sunday"

str2 = "saturday"

print (editDistance(str1, str2, len(str1), len(str2)))

30.