1) **Reverse**a string **without affecting special characters**  
 Given a string **S**, containing special characters and all the alphabets, reverse the string without affecting the positions of the special characters.  
**Input:**A&B  
**Output:**B&A  
**Explanation**: As we ignore '&' and  
As we ignore '&' and then reverse, so answer is "B&A".

PROGRAM:

def reverse\_string(s):

s = list(s)

l, r = 0, len(s) - 1

while l < r:

if not s[l].isalpha():

l += 1

elif not s[r].isalpha():

r -= 1

else:

s[l], s[r] = s[r], s[l]

l += 1

r -= 1

return ''.join(s)

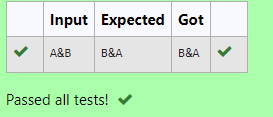
# Test Cases

input\_str=input()

ouput\_str=reverse\_string(input\_str)

print(ouput\_str)

OUTPUT:



2) Given a string S which is of the format USERNAME@DOMAIN.EXTENSION, the program must print the EXTENSION, DOMAIN, USERNAME in the reverse order.

**Input Format:**

The first line contains S.

**Output Format:**

The first line contains EXTENSION.  
The second line contains DOMAIN.  
The third line contains USERNAME.

**Boundary Condition:**

1 <= Length of S <= 100

**Example Input/Output 1:**

Input:

abcd@gmail.com

Output:

com  
gmail  
abcd

PROGRAM:

email=input()

dot=email.index('.')

at=email.index('@')

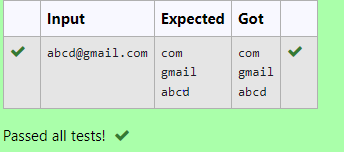
# dot=email.index('.')

print(email[dot+1:])

print(email[at+1:dot])

print(email[:at])

OUTPUT:



3) Write a program to check if two strings are balanced. For example, strings s1 and s2 are balanced if all the characters in the s1 are present in s2. The character’s position doesn’t matter. If balanced display as "true" ,otherwise "false".

PROGRAM:

def are\_strings\_balanced(s1, s2):

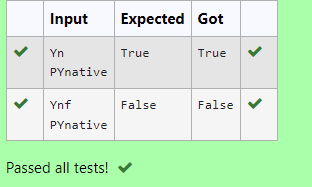
return set(s1).issubset(set(s2))

s1 = input()

s2 = input()

print(are\_strings\_balanced(s1, s2))

OUTPUT:



4) Write a python to read a sentence and print its longest word and its length

PROGRAM:

sentence = input()

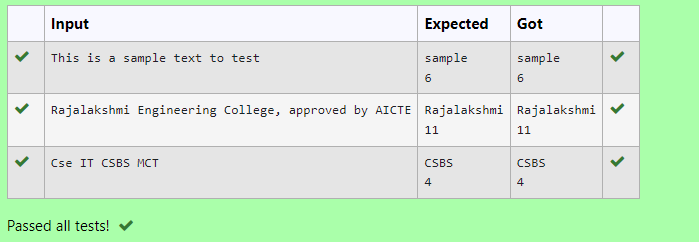
words = sentence.split()

longest\_word = max(words, key=len)

print( longest\_word)

print( len(longest\_word))

OUTPUT:



5) Write a program that takes as input a string (sentence), and returns its second word in uppercase.

For example:

If input is “Wipro Technologies Bangalore” the function should return “TECHNOLOGIES”

If input is “Hello World” the function should return “WORLD”

If input is “Hello” the program should return “LESS”

NOTE 1: If input is a sentence with less than 2 words, the program should return the word “LESS”.

NOTE 2: The result should have no leading or trailing spaces.

PROGRAM:

def extract\_second\_word(sentence):

words = sentence.split()

if len(words) < 2:

return "LESS"

return words[1].upper()

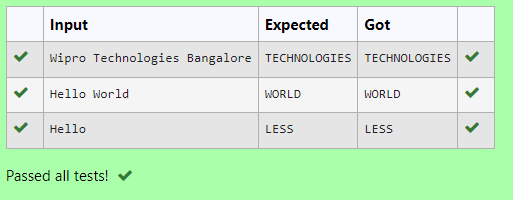
# Test the function

input\_sentence = input()

result = extract\_second\_word(input\_sentence)

print(result)

OUTPUT:



6) Given two Strings s1 and s2, remove all the characters from s1 which is present in s2.

**Constraints**

1<= string length <= 200

**Sample Input 1**

experience

enc

**Sample Output 1**

xpri

PROGRAM:

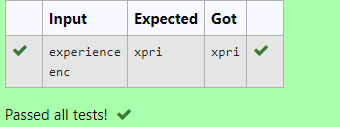
s1 = input()

s2 = input()

result = ''.join([char for char in s1 if char not in s2])

print(result)

OUTPUT:



7) String should contain only the words are not palindrome.

**Sample Input 1**

Malayalam is my mother tongue

**Sample Output 1**

is my mother tongue

PROGRAM:

a=input().lower()

a=a.split()

s=[]

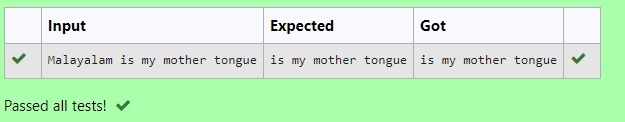
for i in a:

if i!=i[::-1]:

s.append(i)

print(' '.join(s))

OUTPUT:



8) Robert  is having 2 strings consist of uppercase & lowercase english letters. Now he want to compare those two strings lexicographically. The letters' case does not matter, that is an uppercase letter is considered equivalent to the corresponding lowercase letter.

Input

The first line contains **T**. Then **T** test cases follow.

Each test case contains a two lines contains a string. The strings' lengths range from 1 to 100 inclusive. It is guaranteed that the strings are of the same length and also consist of uppercase and lowercase Latin letters.

Output

If the first string is less than the second one, print "-1".  
If the second string is less than the first one, print "1".  
If the strings are equal, print "0".  
Note that the letters' case is not taken into consideration when the strings are compared.

Constraints

**1**≤**T**≤**50**

**String length**≤**100**

PROGRAM:

T = int(input())

for \_ in range(T):

str1 = input().lower()

str2 = input().lower()

if str1 < str2:

print("-1")

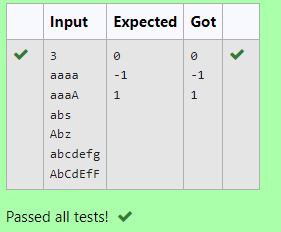
elif str1 > str2:

print("1")

else:

print("0")

OUTPUT:



9)Write a python program to count all letters, digits, and special symbols respectively from a given string

PROGRAM:

def count\_chars(input\_string):

letters = 0

digits = 0

special\_symbols = 0

for char in input\_string:

if char.isalpha():

letters += 1

elif char.isdigit():

digits += 1

else:

special\_symbols += 1

return letters, digits, special\_symbols

input\_string = input()

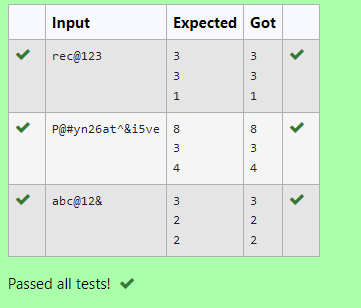
letters, digits, special\_symbols = count\_chars(input\_string)

print(letters)

print(digits)

print(special\_symbols)

OUTPUT:



10) Find if a String2 is substring of String1. If it is, return the index of the first occurrence. else return -1.

**Sample Input 1**

thistest123string

123

**Sample Output 1**

8

PROGRAM:

def find\_substring(string1, string2):

if string2 in string1:

return string1.index(string2)

else:

return -1

# Sample Input

string1 = input()

string2 = input()

# Sample Output

print(find\_substring(string1, string2))

OUTPUT:

