### **5.Experiments based on strings and its operatiom**

**Ex. No. : 5.1 Date: 13/05/2024**

**Register No.: 231801047 Name: HANNAH JAMES**

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:**

h=input("")

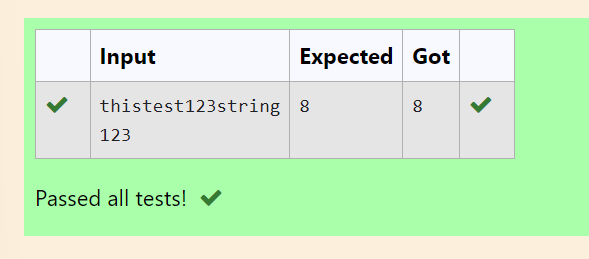
j=input("")

if j in h:

print(h.index(b))

else:

print("-1")



**Ex. No. : 5.2 Date: 13/05/2024**

**Register No.: 231801047 Name: HANNAH JAMES**

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:

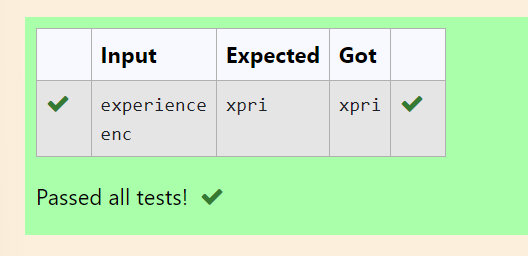
a=input("")

b=input("")

for i in range (len(a)):

if a[i] not in b:

print(a[i],end="")



**Ex. No. : 5.3 Date: 13/05/2024**

**Register No.: 231801047 Name: HANNAH JAMES**

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:

s=input()

at=s.index('@')

d=s.index('.')

un=s[:at]

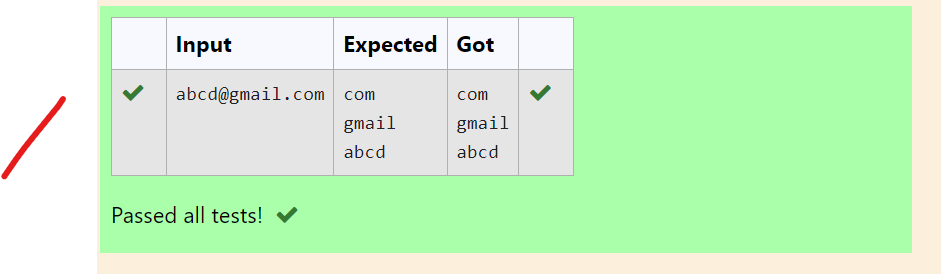
do=s[at+1:d]

ex=s[d+1:]

print(ex)

print(do)

print(un)



**Ex. No. : 5.4 Date: 13/05/2024**

**Register No.: 231801047 Name: HANNAH JAMES**

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:

s=input()

w=s.split()

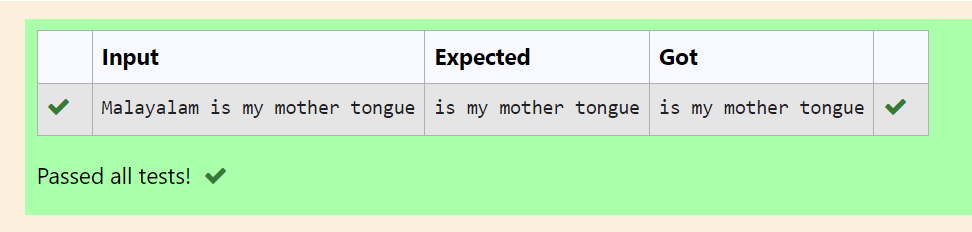
x=''

for wo in w:

wo=wo.lower()

if(wo!=wo[::-1]):

print(wo,end=" ")



**Ex. No. : 5.5 Date: 13/05/2024**

**Register No.: 231801047 Name: HANNAH JAMES**

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".

**For example:**

| **Input** | **Result** |
| --- | --- |
| Yn  PYnative | True |

PROGRAM:

h=input()

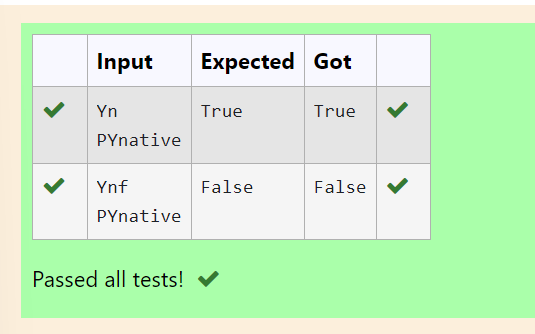
j=input()

if h in j:

print("True")

else:

print("False")



**Ex. No. : 5.6 Date: 13/05/2024**

**Register No.: 231801047 Name: HANNAH JAMES**

Assume that the given string has enough memory.

Don't use any extra space(IN-PLACE)

**Sample Input 1**

a2b4c6

**Sample Output 1**

aabbbbcccccc

PROGRAM:

s = input()

op = ''

i=0

while i < len(s):

c=s[i]

i+=1

n=0

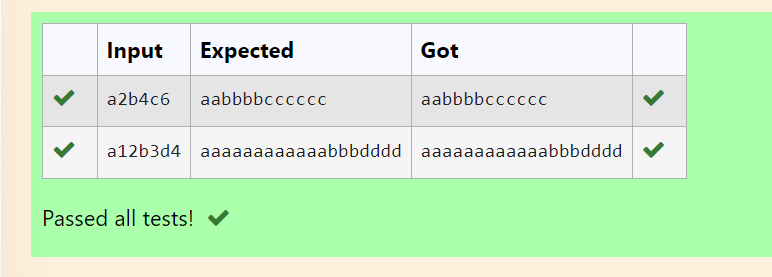
while i<len(s) and s[i].isdigit():

n=n\*10+int(s[i])

i+=1

op += c \* n

print(op)



**Ex. No. : 5.7 Date: 13/05/2024**

**Register No.: 231801047 Name :HANNAH JAMES**

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

**For example:**

| **Input** | **Result** |
| --- | --- |
| rec@123 | 3  3  1 |

PROGRAM:

s=input()

cd=0

ca=0

cs=0

for char in s:

if char.isdigit():

cd += 1

elif char.isalpha():

ca += 1

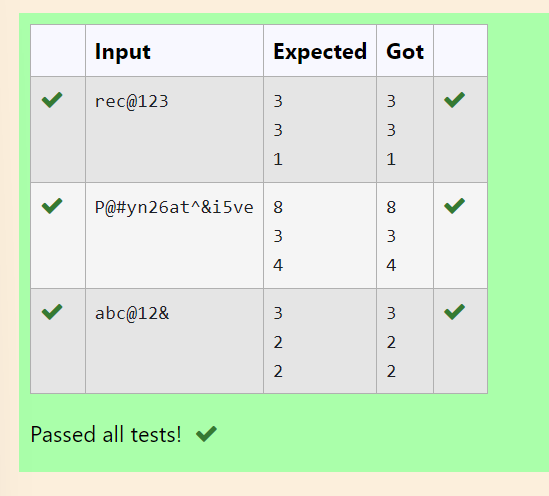
else:

cs += 1

print(ca)

print(cd)

print(cs)



**Ex. No. : 5.8 Date: 13/05/2024**

**Register No.: 231801047 Name: HANNAH JAMES**

In this exercise, you will create a program that reads words from the user until the user enters a blank line. After the user enters a blank line your program should display each word entered by the user exactly once. The words should be displayed in the same order that they were first entered. For example, if the user enters:

first

second

first

third

second

then your program should display:

first

second

third

PROGRAM:

i=1

x=[]

while(i==1):

a=input()

if a==" ":

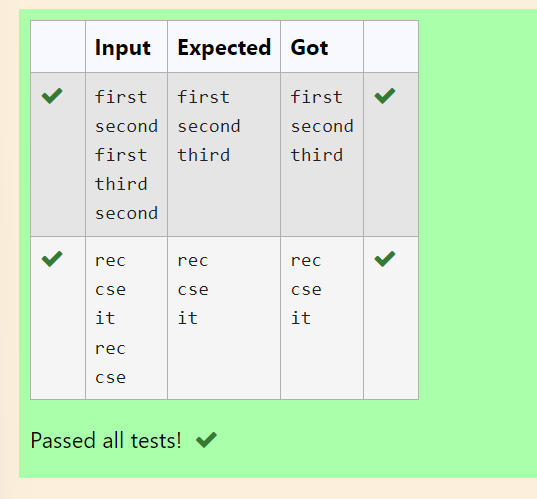
i=0

elif (a!=" " and a not in x):

x.append(a)

for i in x:

print(i)



**Ex. No. : 5.9 Date: 13/05/2024**

**Register No.: 231801047 Name: HANNAH JAMES**

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**

**For example:**

| **Input** | **Result** |
| --- | --- |
| 3  aaaa  aaaA  abs  Abz  abcdefg  AbCdEfF | 0  -1  1 |

PROGRAM:

t = int(input())

for \_ in range(t):

s1 = input().lower()

s2 = input().lower()

if s1<s2:

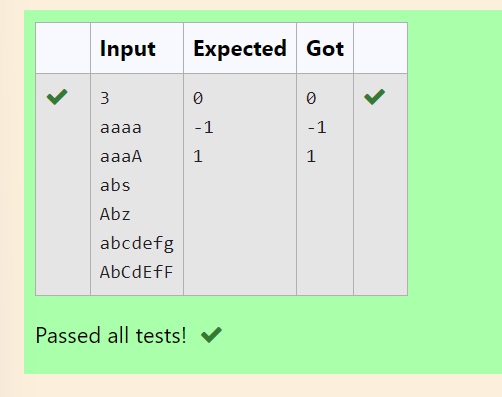
print("-1")

elif s1>s2:

print("1")

else:

print("0")



**Ex. No. : 5.10 Date: 13/05/2024**

**Register No.: 231801047 Name: HANNAH JAMES**

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

**For example:**

| **Input** | **Result** |
| --- | --- |
| This is a sample text to test | sample  6 |

PROGRAM:

def f\_l(s):

w=s.split()

lw=""

ml=0

for wo in w:

if len(wo)>ml:

ml=len(wo)

lw=wo

return lw,ml

s=input()

lw,l=f\_l(s)

print(lw)

print(l)

