1. 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:**

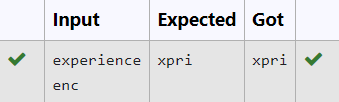
s1=input()

s2=input()

x=''.join(char for char in s1 if char not in s2)

print(x)

**Output:**



2. 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.

**For example:**

| **Input** | **Result** |
| --- | --- |
| Wipro Technologies Bangalore | TECHNOLOGIES |
| Hello World | WORLD |
| Hello | LESS |

**Program:**

s1=input()

x=s1.split()

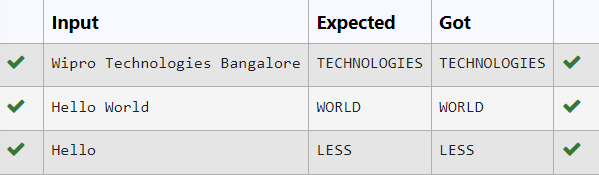
if(len(x)>=2):

print(x[1].upper())

else:

print("LESS")

**Output:**



3. 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:**

sen=input()

words=sen.split()

l=""

maxi=0

for word in words:

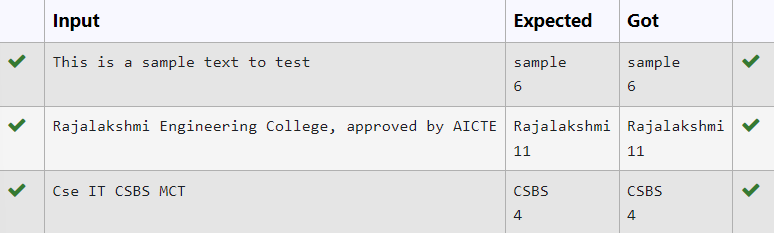
if(len(word)>maxi):

l=word

maxi=len(word)

print(l,maxi,sep="\n")

**Output:**

****

4. 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:**

s1=input()

s2=input()

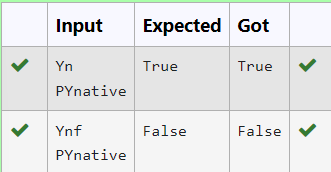
if s1 in s2:

print("True")

else:

print("False")

**Output:**

****

**5.** Given a string S, which contains several words, print the count C of the words whose length is atleast L. (You can include punctuation marks like comma, full stop also as part of the word length. Space alone must be ignored)

**Input Format:**

The first line contains S.  
The second line contains L.

**Output Format:**

The first line contains C

**Boundary Conditions:**

2 <= Length of S <= 1000

**Example Input/Output 1:**

Input:

During and after Kenyattas inauguration police elsewhere in the capital, Nairobi, tried to stop the opposition from holding peaceful demonstrations.  
5

Output:

13

Explanation:

The words of minimum length 5 are  
During  
after  
Kenyattas  
inauguration  
police  
elsewhere  
capital,  
Nairobi,  
tried  
opposition  
holding  
peaceful  
demonstrations.

**Program:**

s=input()

L=int(input())

words=s.split()

c=0

for word in words:

if len(word)>=L:

c=c+1

print(c)

**Output:**

|  | **Input** | **Expected** | **Got** |  |
| --- | --- | --- | --- | --- |
|  | During and after Kenyattas inauguration police elsewhere in the capital, Nairobi, tried to stop the opposition from holding peaceful demonstrations.  5 | 13 | 13 |  |

6. Two string values S1, S2 are passed as the input. The program must print first N characters present in S1 which are also present in S2.

**Input Format:**

The first line contains S1.  
The second line contains S2.  
The third line contains N.

**Output Format:**

The first line contains the N characters present in S1 which are also present in S2.

**Boundary Conditions:**

2 <= N <= 10  
2 <= Length of S1, S2 <= 1000

**Example Input/Output 1:**

Input:

abcbde  
cdefghbb  
3

Output:

bcd

**Note:**

b occurs twice in common but must be printed only once.

**Program:**

s1=input()

s2=input()

N=int(input())

s2set=set(s2)

cc=[]

c=0

for char in s1:

if char in s2set and char not in cc:

cc.append(char)

c=c+1

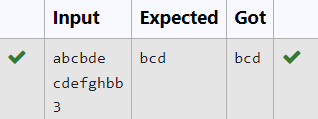
if c==N:

break

x=''.join(cc)

print(x)

**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:**

s=input()

words=s.split()

x=''

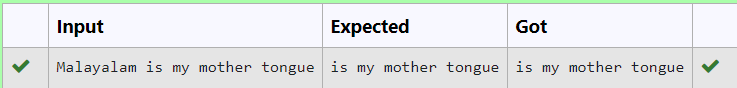
for word in words:

word=word.lower()

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

print(word,end=" ")

**Output:**

****

8. 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:**

s1=input()

r=""

i=0

while i< len(s):

char=s[i]

i+=1

num=""

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

num+=s[i]

i+=1

r+=char\*int(num)

print(r)

**Output:**

9. **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".

**For example:**

| **Input** | **Result** |
| --- | --- |
| A&x# | x&A# |

**Program:**

s=input()

l=[]

for i in s:

if(i.isalpha()):

l.append(i)

l.reverse()

r=''

index=0

for i in s:

if(i.isalpha()):

r+=l[index]

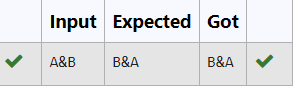
index+=1

else:

r+=i

print(r)

**Output:**

****

**10.** 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

**For example:**

| **Input** | **Result** |
| --- | --- |
| arvijayakumar@rajalakshmi.edu.in | edu.in  rajalakshmi  arvijayakumar |

**Program:**

s=input()

at=s.index('@')

dot=s.index('.')

username=s[:at]

domain=s[at+1:dot]

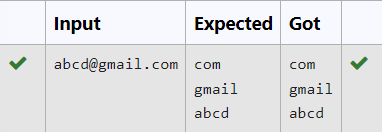
exten=s[dot+1:]

print(exten)

print(domain)

print(username)

**Output:**

****