

|  |  |
| --- | --- |
| **06** | **- Strings in Python** |
|  | . 104 |



|  |  |  |  |
| --- | --- | --- | --- |
| **Ex. No.** | **:** | **6.1** | **Date:** 13/4/24 |
| **Register No.:231501044** | | | **Name: Edmond Allan A** |
|  |  |  |  |

**Count Chars**

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**

x = str(input())

c1, c2, c3 = 0, 0, 0

for i in x:

if i.isalpha():

c1 += 1

elif i.isdigit():

c2 += 1

else:

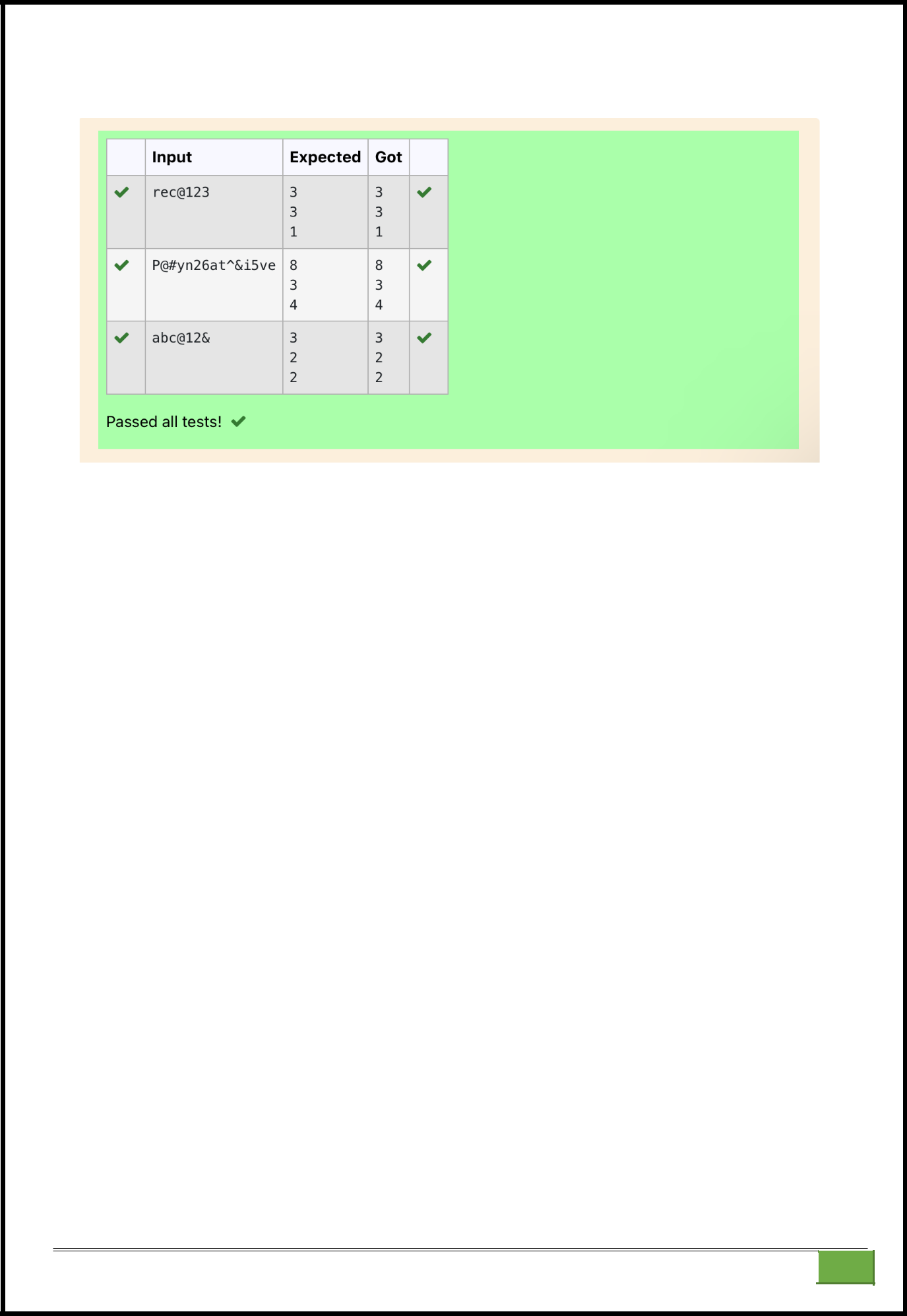
c3 += 1

print(c1)

print(c2)

print(c3)

. 105

**Output:**

. 106

|  |  |  |  |
| --- | --- | --- | --- |
| **Ex. No.** | **:** | **6.2** | **Date:** 13/4/24 |
| **Register No.:231501044** | | | **Name: Edmond Allan A** |
|  |  |  |  |



**Decompress the String**

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()

l1=[]

l2=[]

s2=""

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

if((ord(s[i])>=65 and ord(s[i])<=90) or (ord(s[i])>=97 and ord(s[i])<=122)):

l1.append(s[i])

else:

s2=s2+s[i]

continue

if s2!="":

l2.append(s2)

s2=""

i=-1

while(s[i]==ord(s[i])>=48 and ord(s[i])<=57):

s2=s2+s[i]

i-=1

continue

l2.append(s2)

'''print(l1,l2)'''

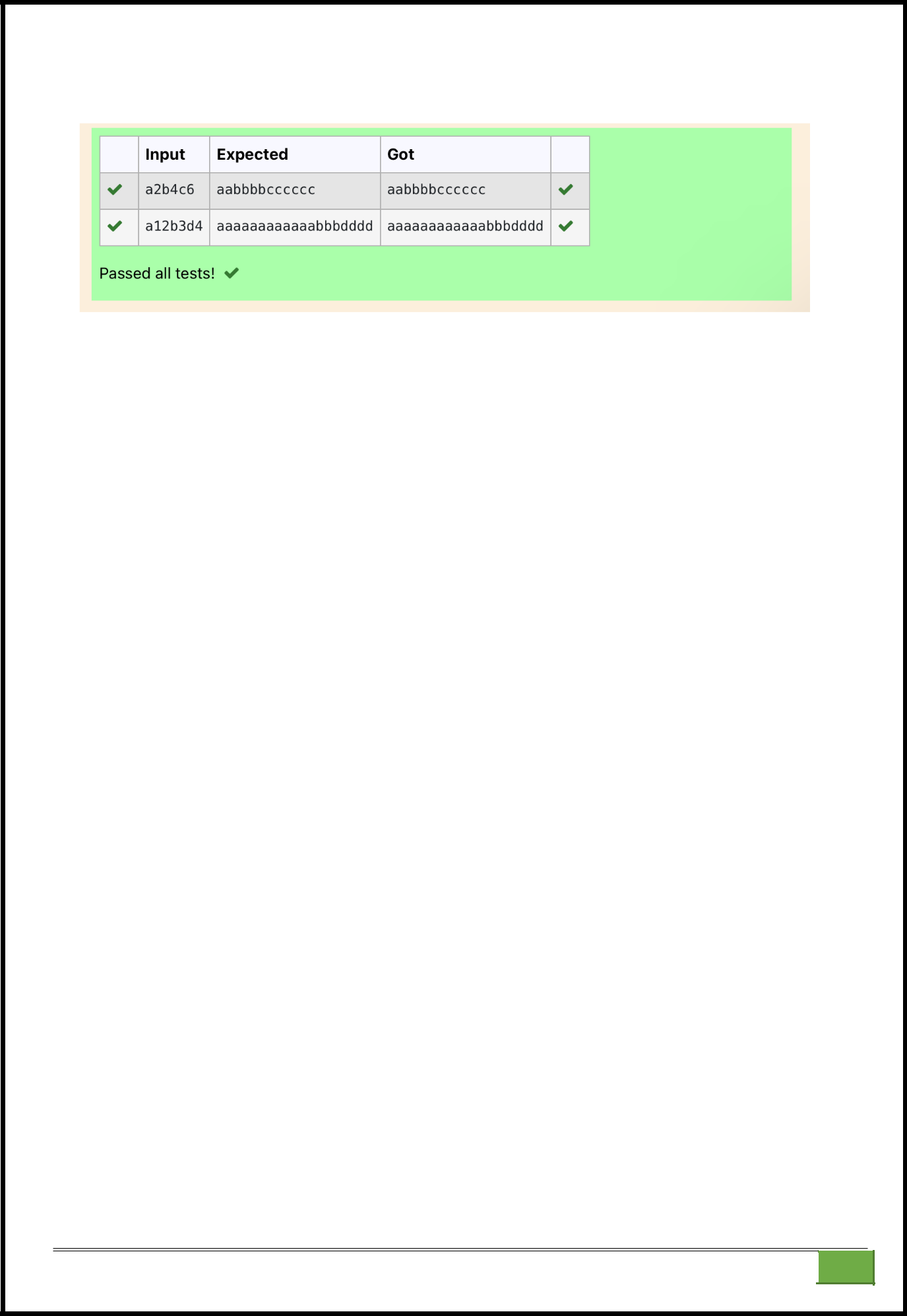
for i in range(0,len(l2)):

l2[i]=int(l2[i])

for i in range(0,len(l1)):

print(l1[i]\*l2[i],end="")

. 107

**Output:**

. 108



|  |  |  |  |
| --- | --- | --- | --- |
| **Ex. No.** | **:** | **6.3** | **Date:** 13/4/24 |
| **Register No.:231501044** | | | **Name: Edmond Allan A** |
|  |  |  |  |

**First N Common Chars**

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().strip()

s2 = input().strip()

n = int(input())

common\_chars = ''

for char in s1:

if char in s2 and char not in common\_chars:

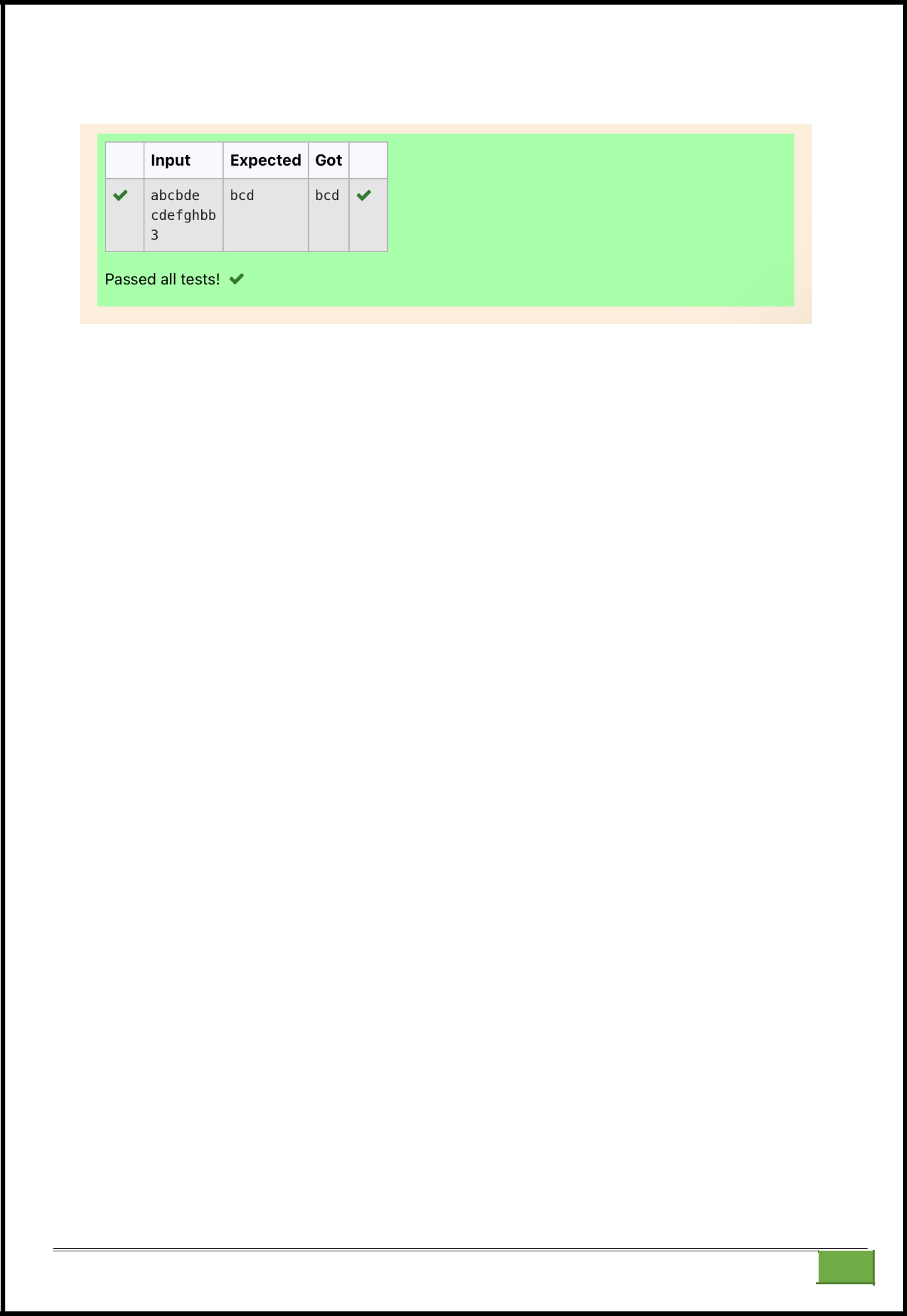
common\_chars += char

if len(common\_chars) == n:

break

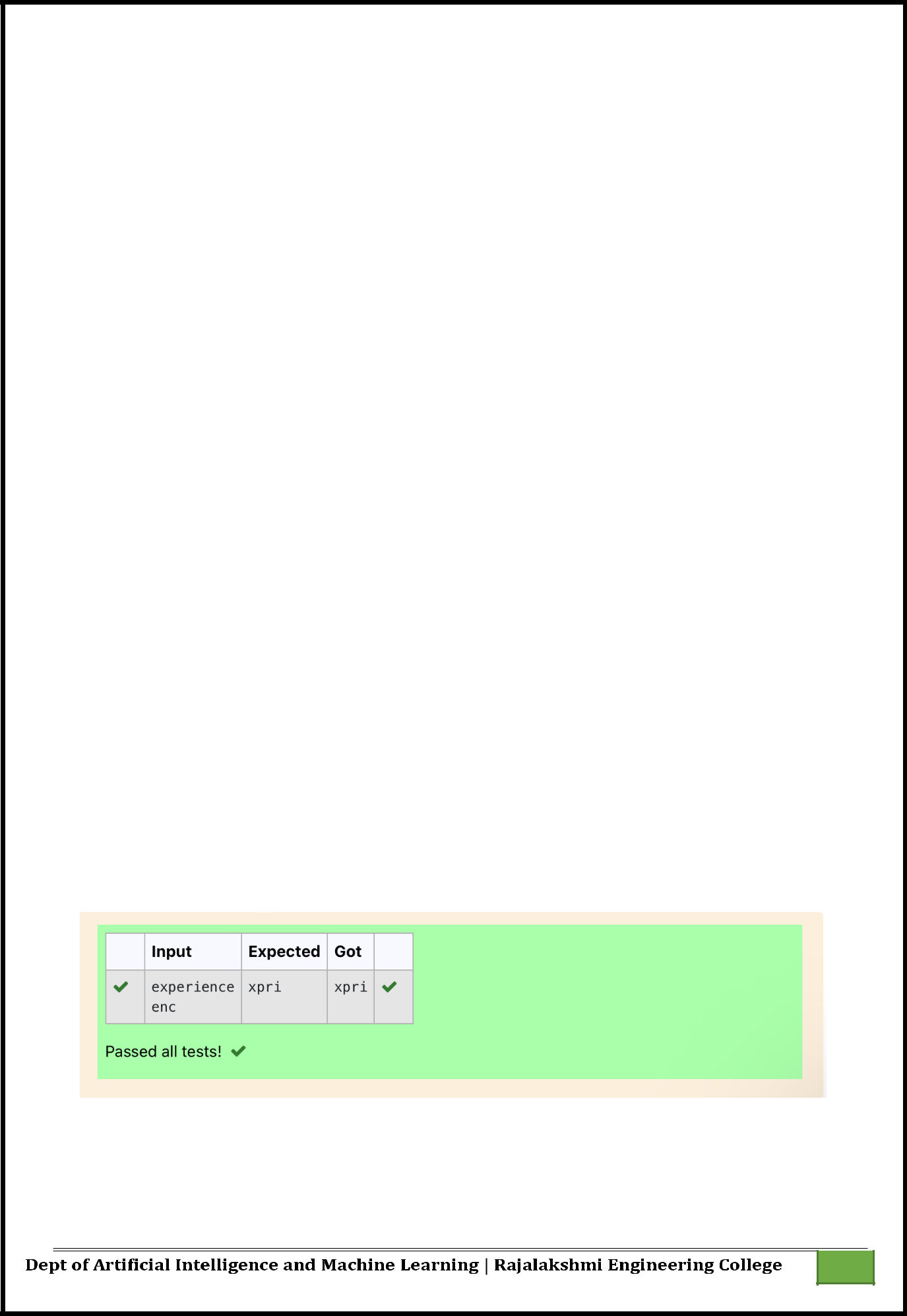
print(common\_chars)

. 109

Output:

. 110

|  |  |  |  |
| --- | --- | --- | --- |
| **Ex. No.** | **:** | **6.4** | **Date:** 13/4/24 |
| **Register No.:231501044** | | | **Name: Edmond Allan A** |
|  |  |  |  |



**Remove Characters**

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**

str1=str(input())

str2=str(input())

l1=list(str1)

l2=list(str2)

for i in range(0,len(l2)):

for j in range(0,len(l1)):

if l2[i] in l1:

l1.remove(l2[i])

str3=""

for i in range(0,len(l1)):

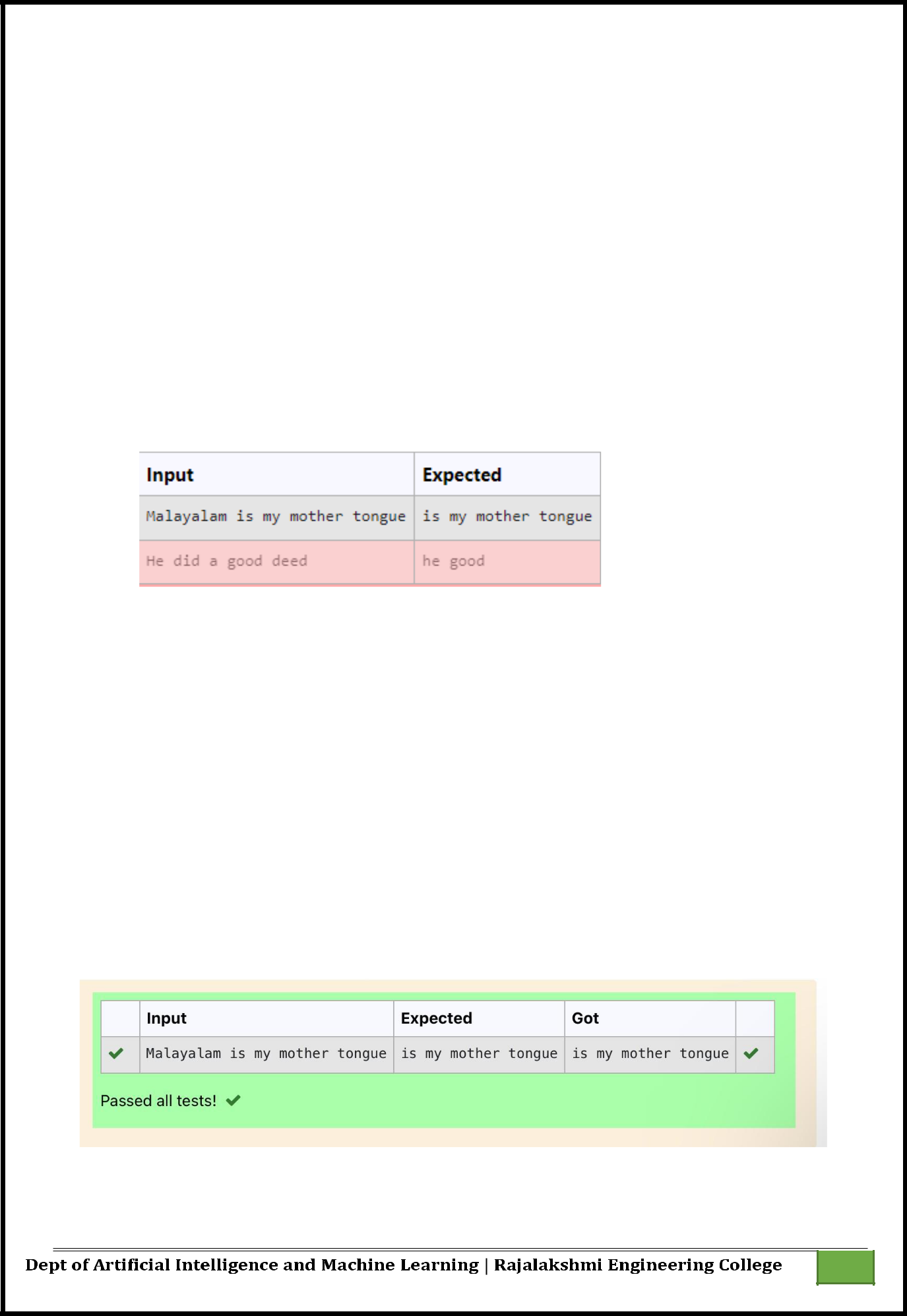
str3=str3+l1[i]

print(str3)

Output:

. 111

|  |  |  |  |
| --- | --- | --- | --- |
| **Ex. No.** | **:** | **6.5** | **Date:** 13/4/24 |
| **Register No.:231501044** | | | **Name: Edmond Allan A** |
|  |  |  |  |



**Remove Palindrome Words**

String should contain only the words are not palindrome.

Sample Input 1

Malayalam is my mother tongue

Sample Output 1

is my mother tongue

For example:

**PROGRAM**

x = input()

words = x.split()

non\_palindrome\_words = []

for word in words:

word\_lower = word.lower()

if word\_lower != word\_lower[::-1]:

non\_palindrome\_words.append(word\_lower)

for word in non\_palindrome\_words:

print(word, end=' ')

Output:

. 112

|  |  |  |  |
| --- | --- | --- | --- |
| **Ex. No.** | **:** | **6.6** | **Date:** 13/4/24 |
| **Register No.:231501044** | | | **Name: Edmond Allan A** |
|  |  |  |  |



**Return Second World in Uppercase**

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

Wipro Technologies Bangalore

Hello World

Hello

Result

TECHNOLOGIES

WORLD

LESS

**PROGRAM**

a=input()

l=a.split()

#print(l)

if(len(l)<2):

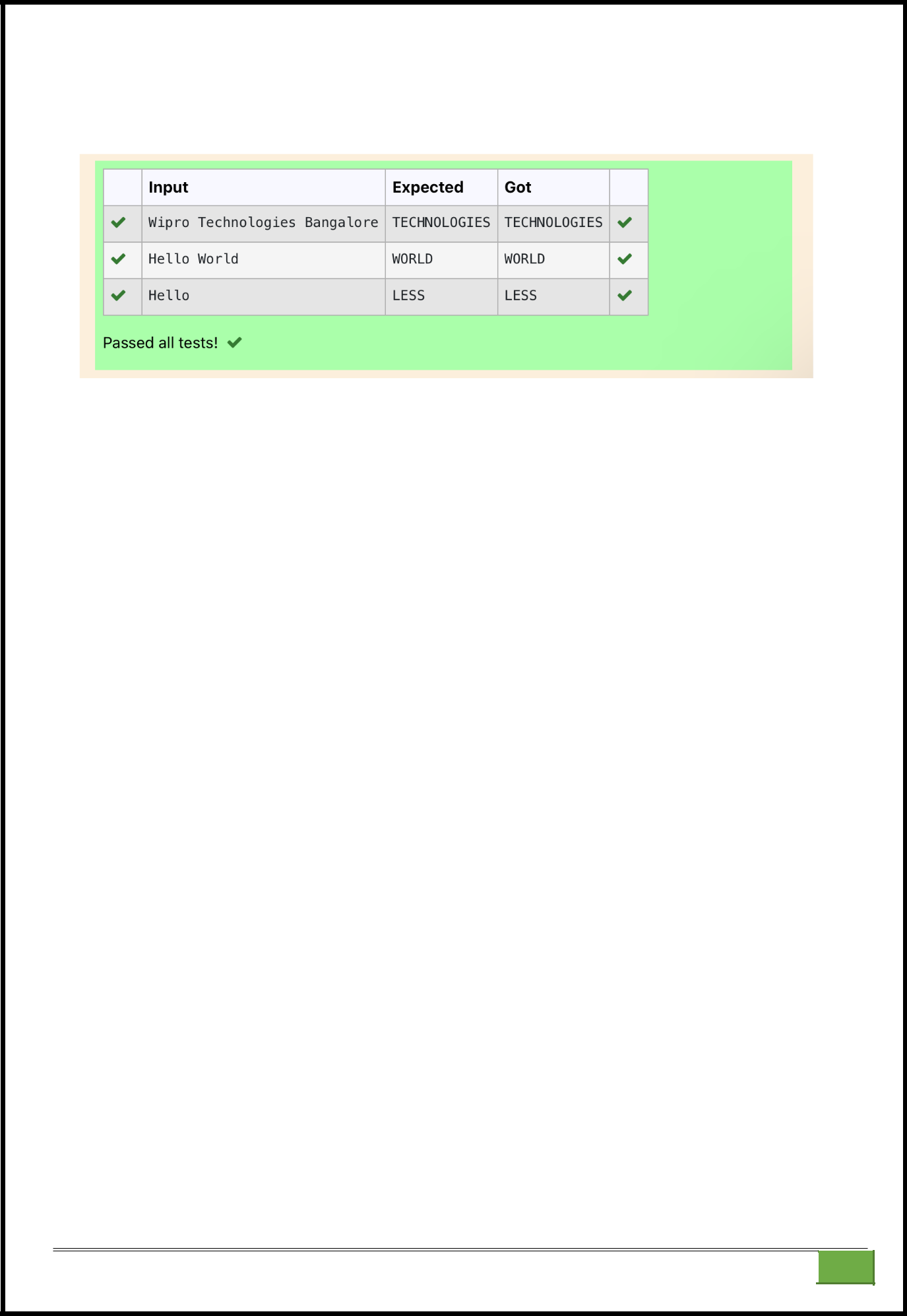
print("LESS")

else:

a=l[1]

print(a.upper())

. 113

**Output:**

. 114

|  |  |  |  |
| --- | --- | --- | --- |
| **Ex. No.** | **:** | **6.7** | **Date:** 13/4/24 |
| **Register No.:231501044** | | | **Name: Edmond Allan A** |
|  |  |  |  |



**Revers String**

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().strip()

left, right = 0, len(s) - 1

s\_list = list(s)

while left < right:

if not s\_list[left].isalpha():

left += 1

elif not s\_list[right].isalpha():

right -= 1

else:

s\_list[left], s\_list[right] = s\_list[right], s\_list[left] left += 1

right -= 1

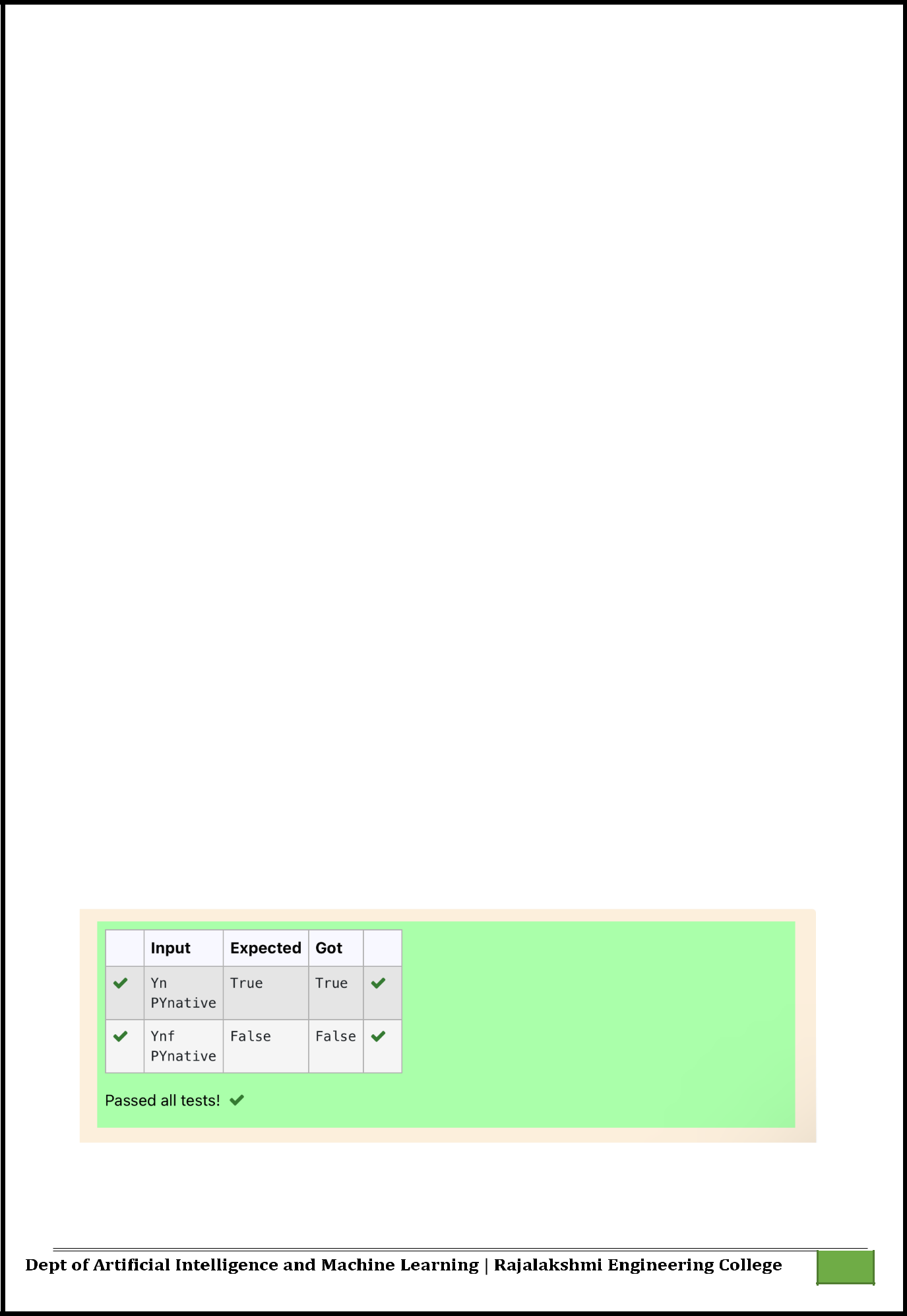
result = ''.join(s\_list)

print(result)

. 115

**Output:**

. 116



|  |  |  |  |
| --- | --- | --- | --- |
| **Ex. No.** | **:** | **6.8** | **Date:** 13/4/24 |
| **Register No.:231501044** | | | **Name: Edmond Allan A** |
|  |  |  |  |

**String characters balance Test**

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**

x = input()

y = input()

if x in y:

print("True")

else:

print("False")

Output:

. 117

|  |  |  |  |
| --- | --- | --- | --- |
| **Ex. No.** | **:** | **6.9** | **Date:** 13/4/24 |
| **Register No.:231501044** | | | **Name: Edmond Allan A** |
|  |  |  |  |



**Unique Names**

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:

**Input:**

first

second

first

third

second

then your program should display:

**Output:**

first

second

third

**PROGRAM**

b = input()

s1 =[]

c = 0

while b != " ":

s1.append(b)

b = input()

my\_set = list(set(s1))

for i in s1:

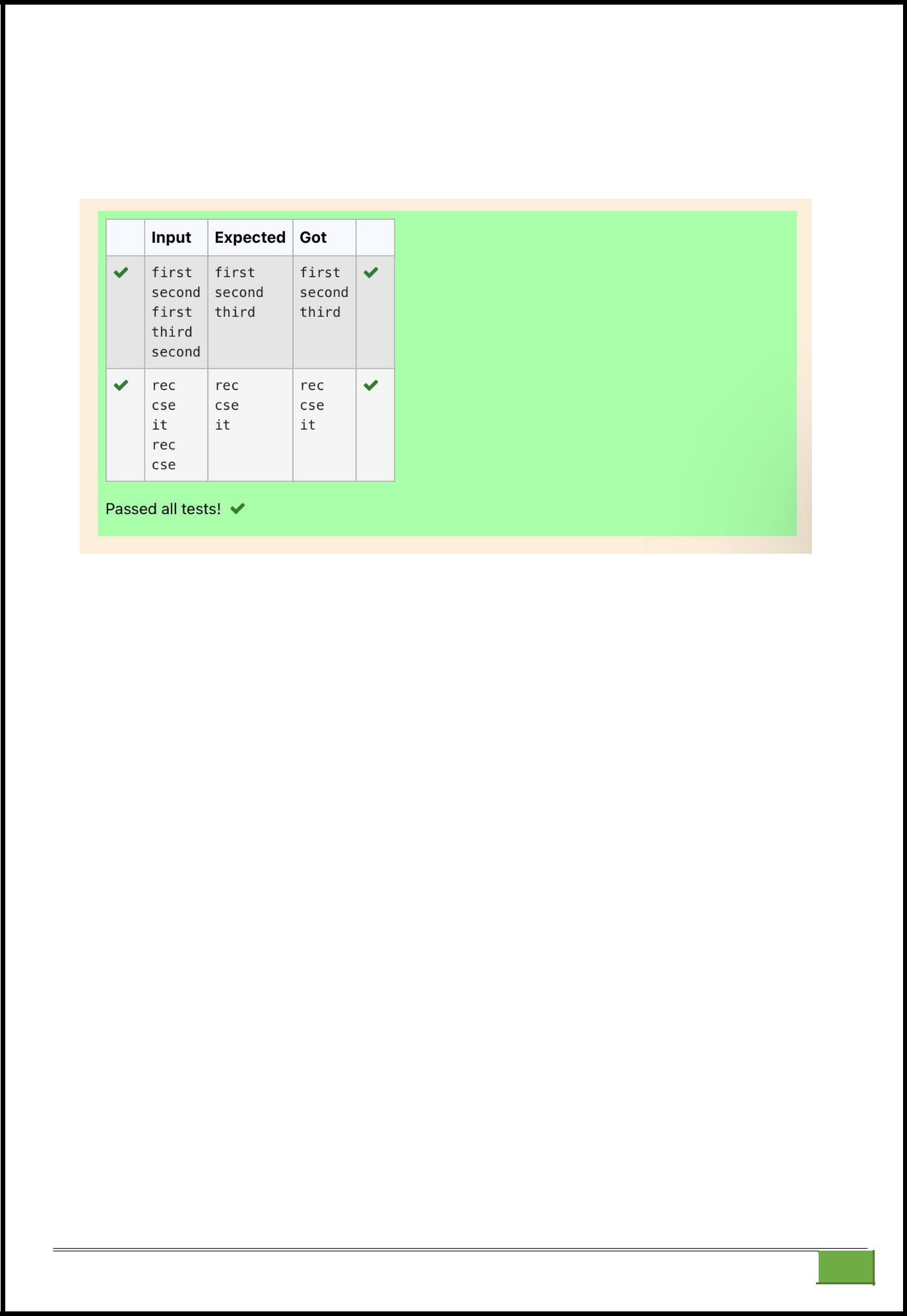
for j in range(len(my\_set)):

if i == my\_set[j]:

print(i)

my\_set[j] = "0000"

. 118

**Output:**

. 119

|  |  |  |  |
| --- | --- | --- | --- |
| **Ex. No.** | **:** | **6.10** | **Date:** 13/4/24 |
| **Register No.:231501044** | | | **Name: Edmond Allan A** |
|  |  |  |  |



**Username Domain Extension**

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**:

vijayakumar.r@rajalakshmi.edu.in

**Output**:

edu.in

rajalakshmi

vijayakumar.r

**PROGRAM**

b=0

c=0

a=input()

if '@' in a:

b=a.index("@")

if "." in a:

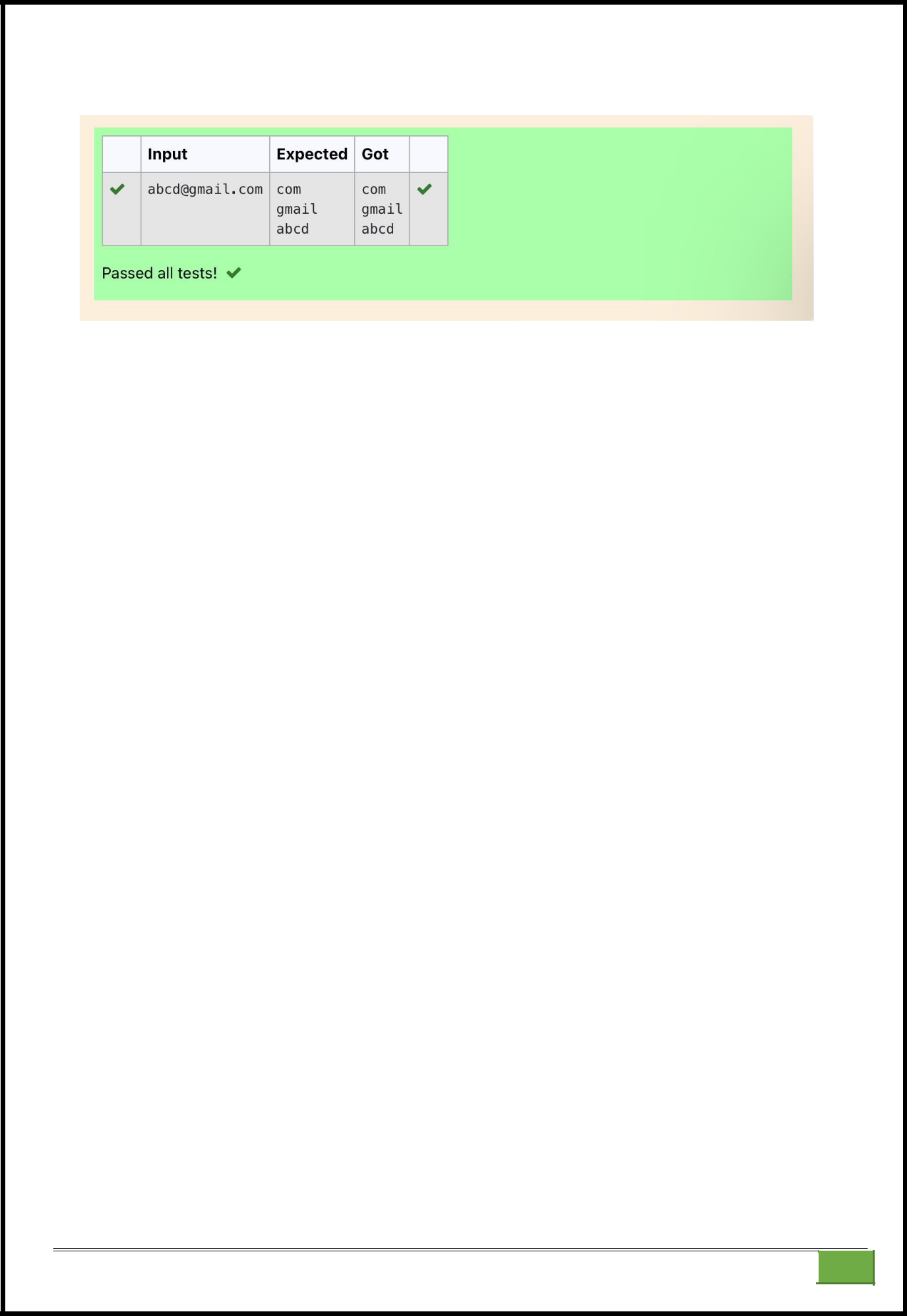
c=a.index(".")

print(a[c+1:])

print(a[b+1:c])

print(a[:b])

. 120

Output:

. 121