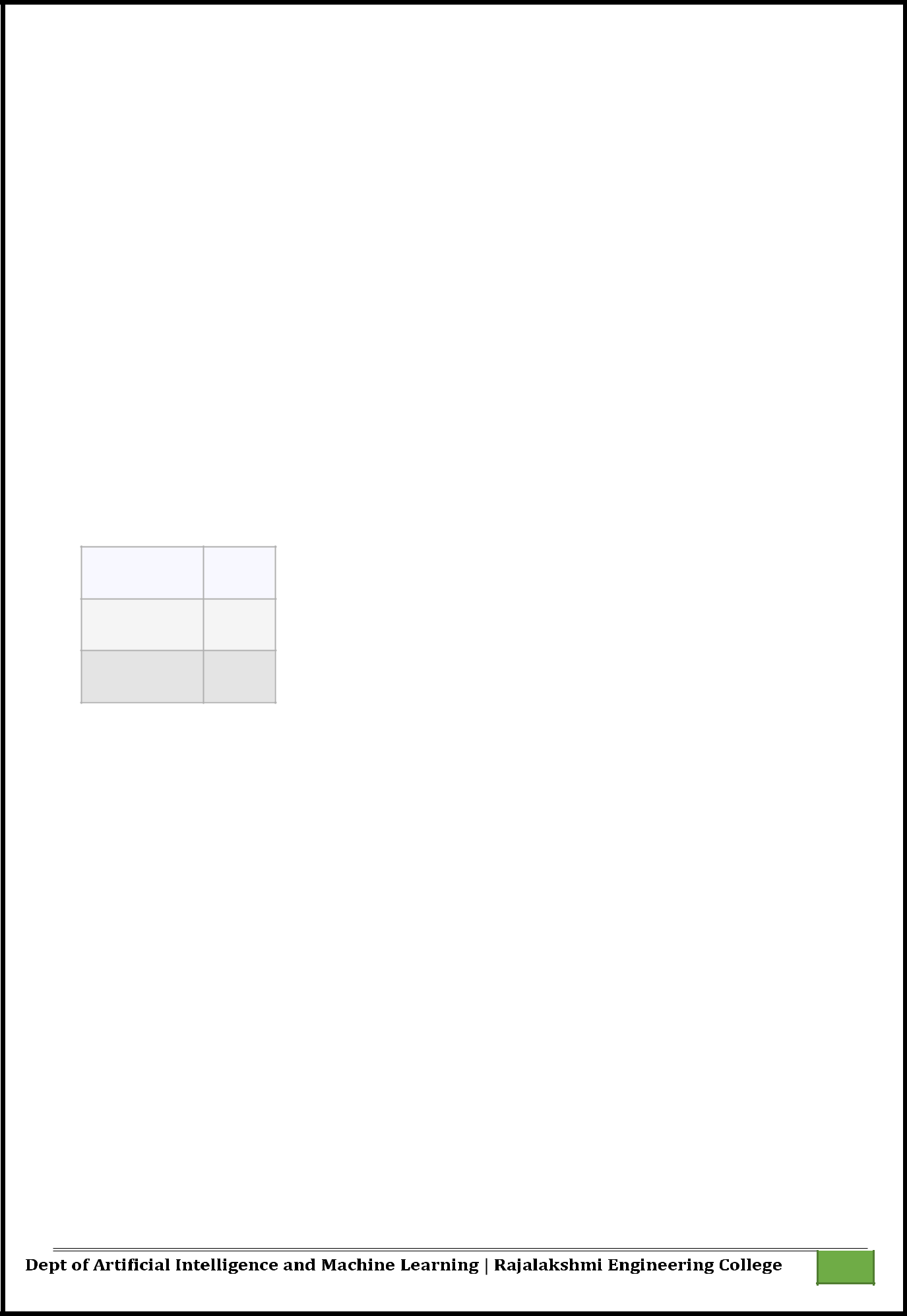


|  |  |
| --- | --- |
| **08** | **– Tuple/Set** |
|  | . 137 |



|  |  |  |  |
| --- | --- | --- | --- |
| **Ex. No.** | **:** | **8.1** | **Date:** 27/4/24 |
| **Register No.:231501044** | | | **Name: Edmond Allan A** |
|  |  |  |  |

**Binary String**

Coders here is a simple task for you, Given string str. Your task is to check whether it is a binary string or not by using python set.

Examples:

Input: str = "01010101010"

Output: Yes

Input: str = "REC101"

Output: No

**For example:**

**Input**

**Result**

01010101010

Yes

010101 10101

No

**PROGRAM**

str=input()

str=[x for x in str]

str=set(str)

c=0

if '0' in str and '1' in str:

c+=2

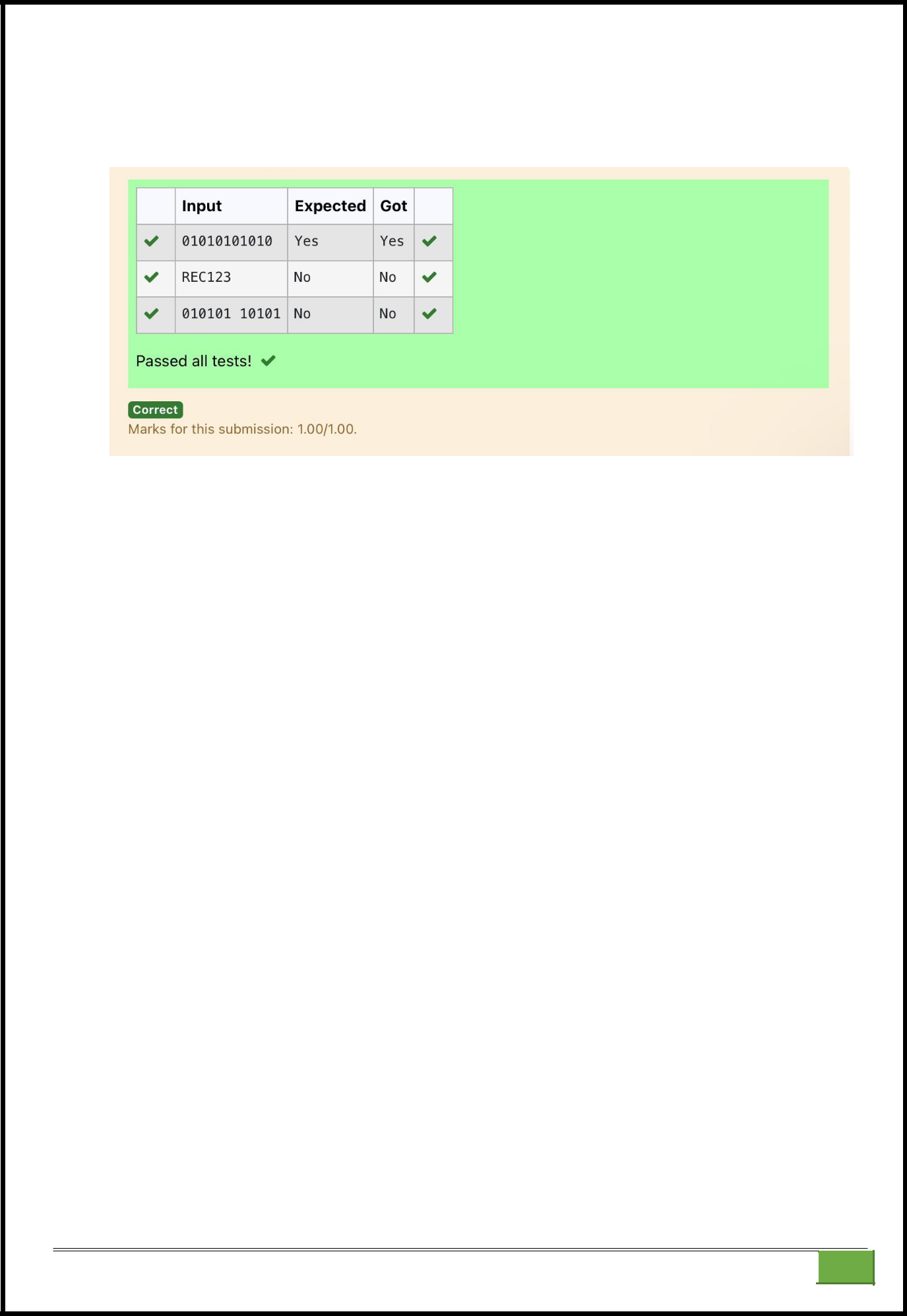
if(c==len(str)):

print("Yes")

else:

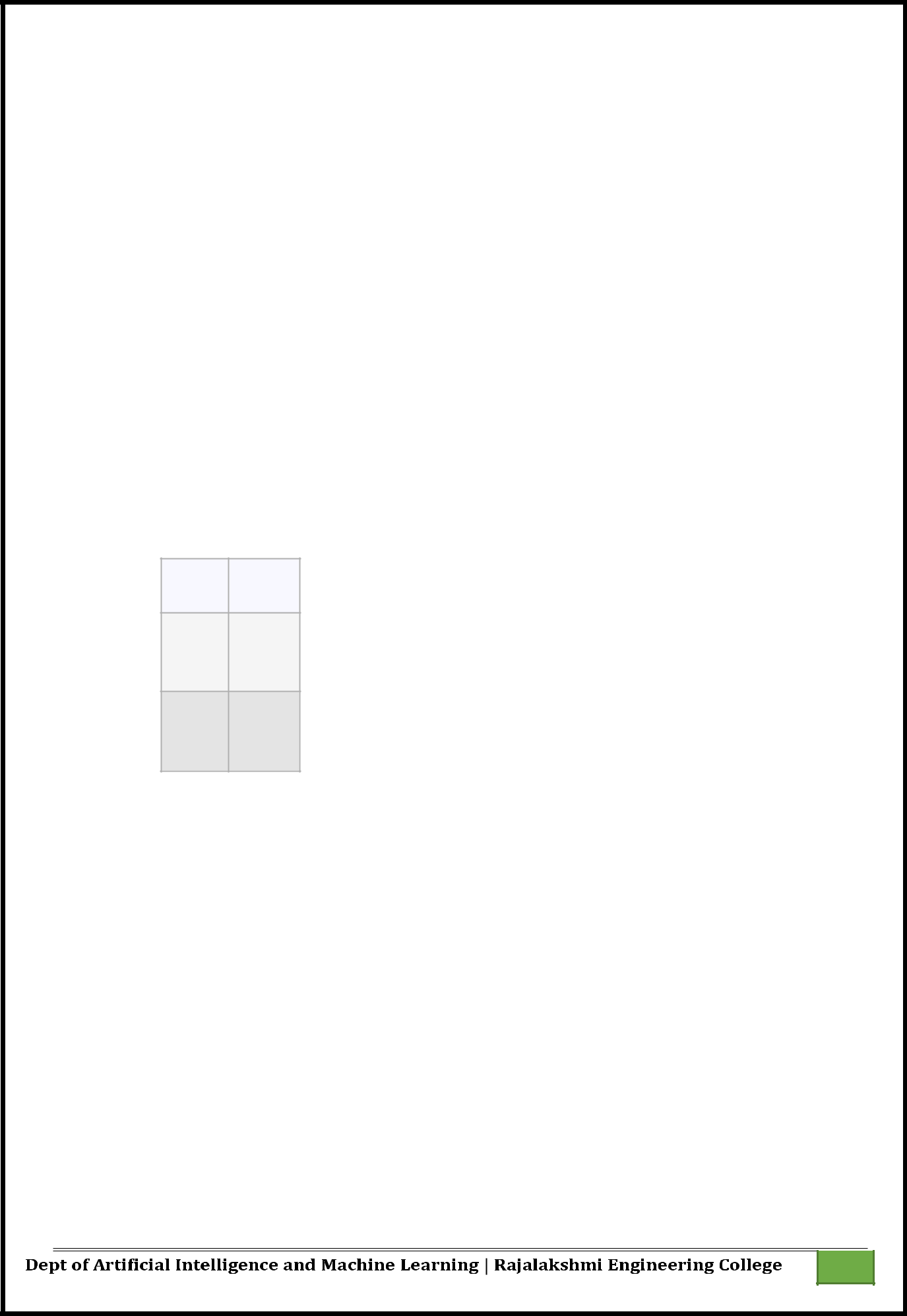
print("No")

. 138

**Output:**

. 139

|  |  |  |  |
| --- | --- | --- | --- |
| **Ex. No.** | **:** | **8.2** | **Date:** 27/4/24 |
| **Register No.:231501044** | | | **Name: Edmond Allan A** |
|  |  |  |  |



**Check Pair**

Given a tuple and a positive integer k, the task is to find the count of distinct pairs in the tuple whose sum is equal to **K**.

Examples**:**

**Input**: t = (5, 6, 5, 7, 7, 8 ), K = 13

**Output**: 2

Explanation:

Pairs with sum K( = 13) are {(5, 8), (6, 7), (6, 7)}.

Therefore, distinct pairs with sum K( = 13) are { (5, 8), (6, 7) }.

Therefore, the required output is 2.

For example:

|  |  |
| --- | --- |
| Input | Result |
| 1,2,1,2,5 | 1 |
| 3 |  |

|  |  |  |  |
| --- | --- | --- | --- |
| 1,2 |  | 0 |  |
|  |  |
|  |  |  |  |

0

**PROGRAM**

s=input()

l1=s.split(',')

n=int(input())

l2=[]

for i in range(len(l1)):

j=i+1

for j in range(len(l1)):

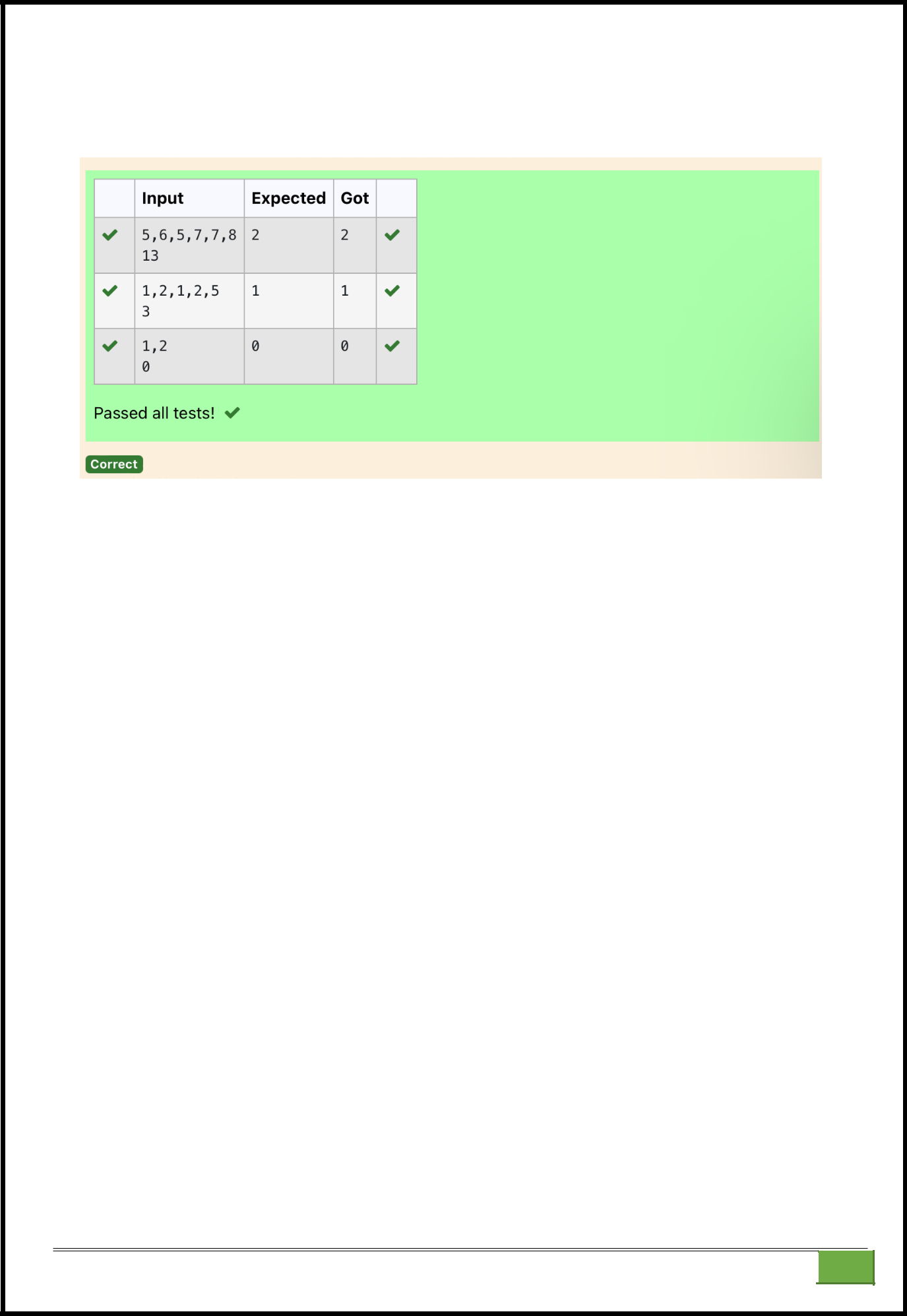
if(int(l1[i])+int(l1[j])==n):

l2.append(list[l1[i],l1[j]])

s=set(l2)

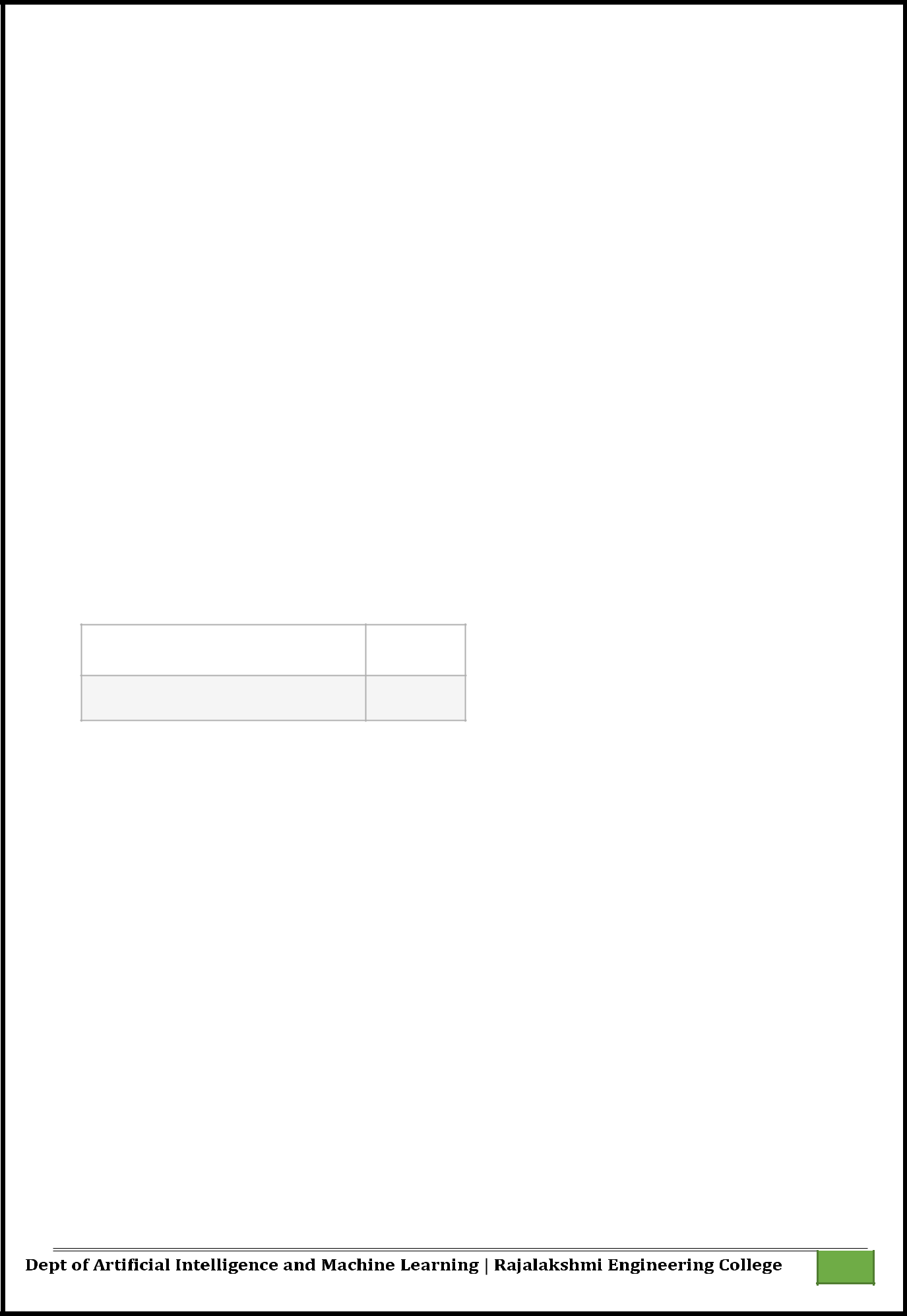
print(len(s)//2)

. 140

Output:

. 141

|  |  |  |  |
| --- | --- | --- | --- |
| **Ex. No.** | **:** | **8.3** | **Date:** 27/4/24 |
| **Register No.:231501044** | | | **Name: Edmond Allan A** |
|  |  |  |  |



**DNA Sequence**

The **DNA sequence** is composed of a series of nucleotides abbreviated as 'A', 'C', 'G', and 'T'.

For example, "ACGAATTCCG" is a **DNA sequence**.

When studying **DNA**, it is useful to identify repeated sequences within the DNA.

Given a string s that represents a **DNA sequence**, return all the **10-letter- long** sequences (substrings) that occur more than once in a DNA molecule. You may return the answer in **any order**.

**Example 1:**

**Input:** s = "AAAAACCCCCAAAAACCCCCCAAAAAGGGTTT"

**Output:** ["AAAAACCCCC","CCCCCAAAAA"]

**Example 2:**

**Input:** s = "AAAAAAAAAAAAA"

**Output:** ["AAAAAAAAAA"]

**For example:**

|  |  |  |
| --- | --- | --- |
| **Input** |  | **Result** |
|  |  |  |

AAAAACCCCCAAAAACCCCCCAAAAAGGGTTT AAAAACCCCC

CCCCCAAAAA

**PROGRAM**

s = input()

sequence\_length = 10

seen = {}

result = []

for i in range(len(s) - sequence\_length + 1):

sequence = s[i:i + sequence\_length]

if sequence in seen:

seen[sequence] += 1

else:

seen[sequence] = 1

for sequence, count in seen.items():

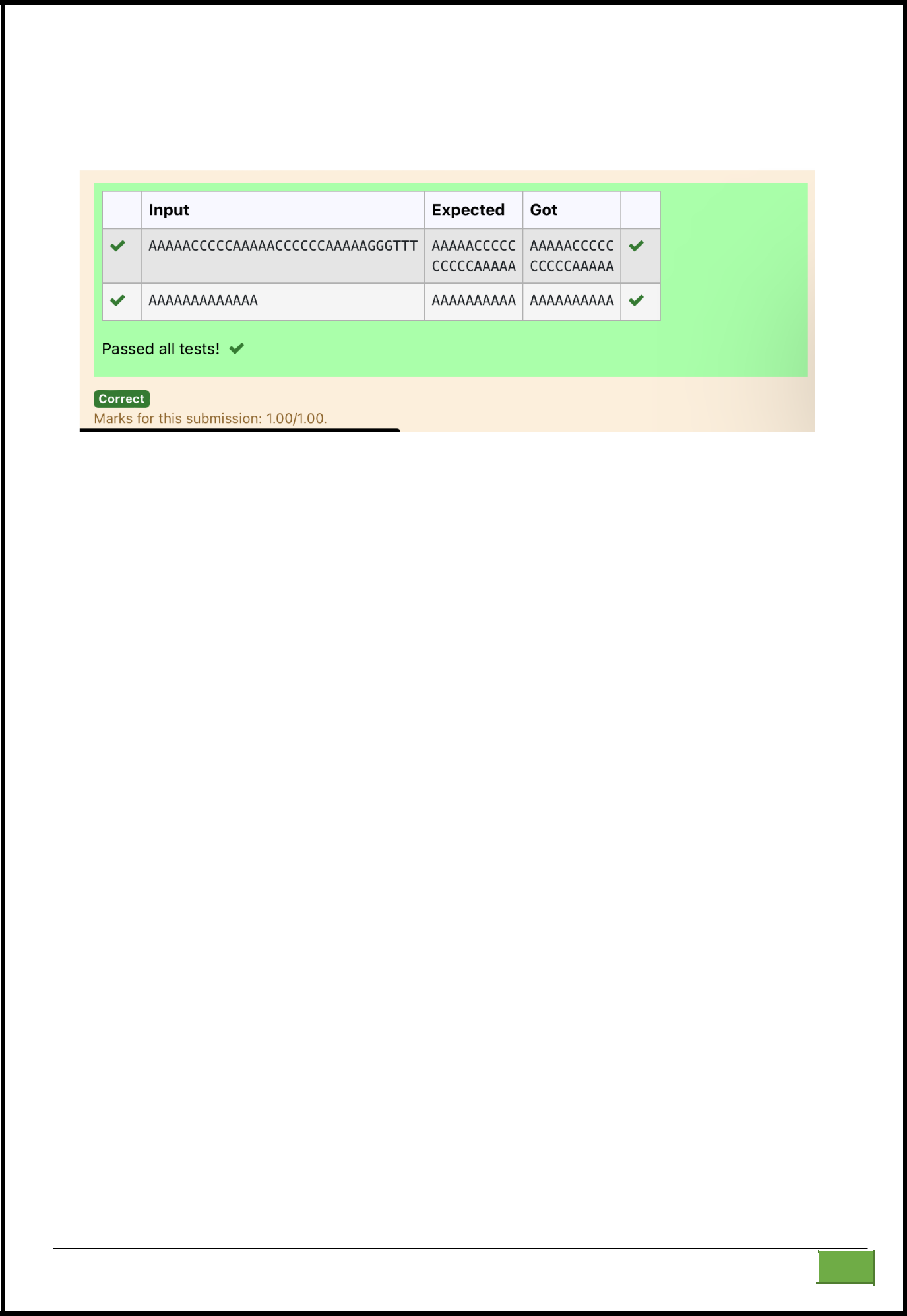
if count > 1:

result.append(sequence)

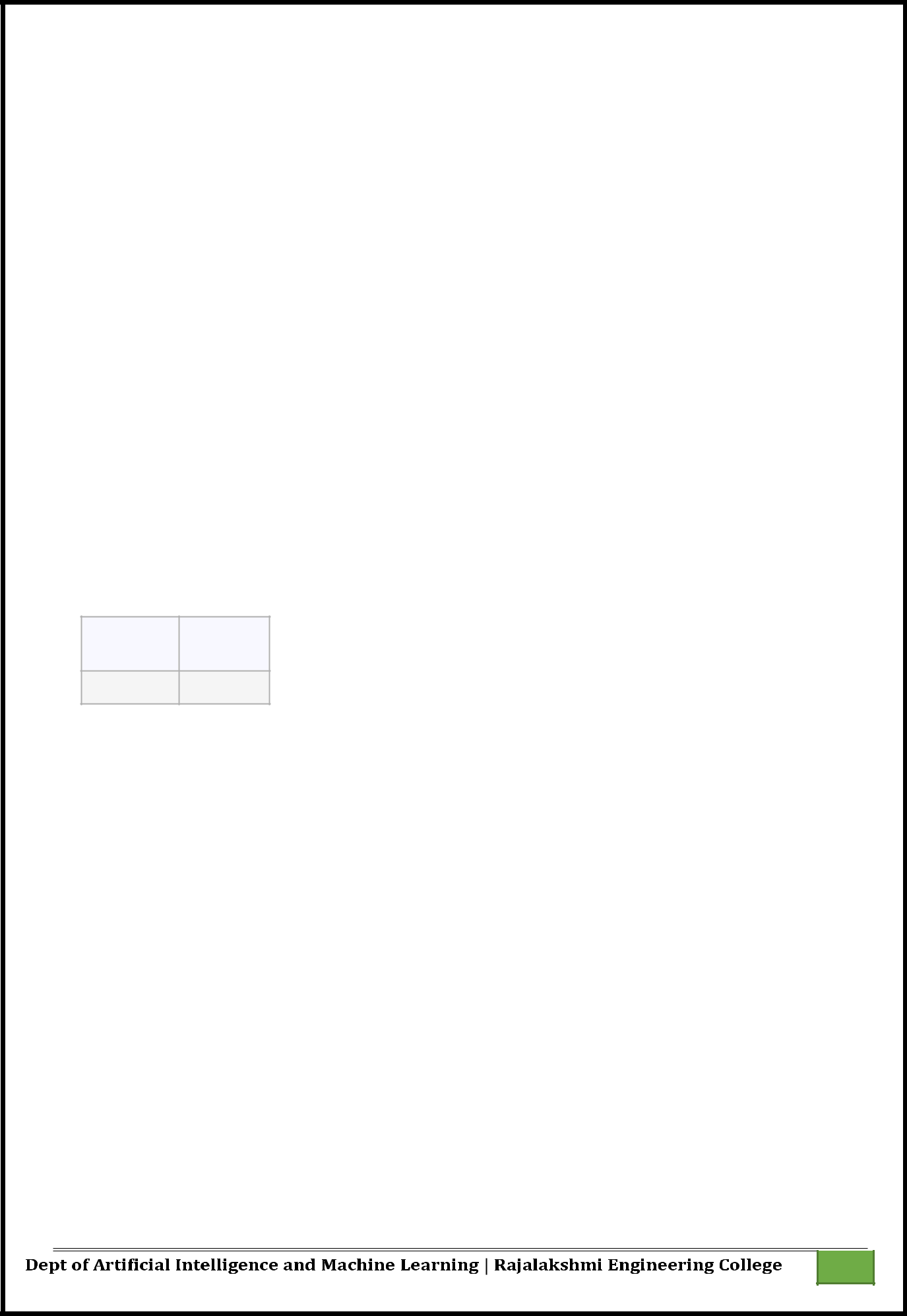
for i in result:

print(i)

. 142

**Output:**

. 143



|  |  |  |  |
| --- | --- | --- | --- |
| **Ex. No.** | **:** | **8.4** | **Date:** 27/4/24 |
| **Register No.:231501044** | | | **Name: Edmond Allan A** |
|  |  |  |  |

**Print repeated no**

Given an array of integers nums containing n + 1 integers where each integer is in the range [1, n] inclusive.There is only **one repeated number** in nums, return *this repeated number*. Solve the problem using set.

**Example 1:**

**Input:** nums = [1,3,4,2,2]

**Output:** 2

**Example 2:**

**Input:** nums = [3,1,3,4,2]

**Output:** 3

**For example:**

**Input**

**Result**

13442

4

**PROGRAM**

st=input()

nums=st.split()

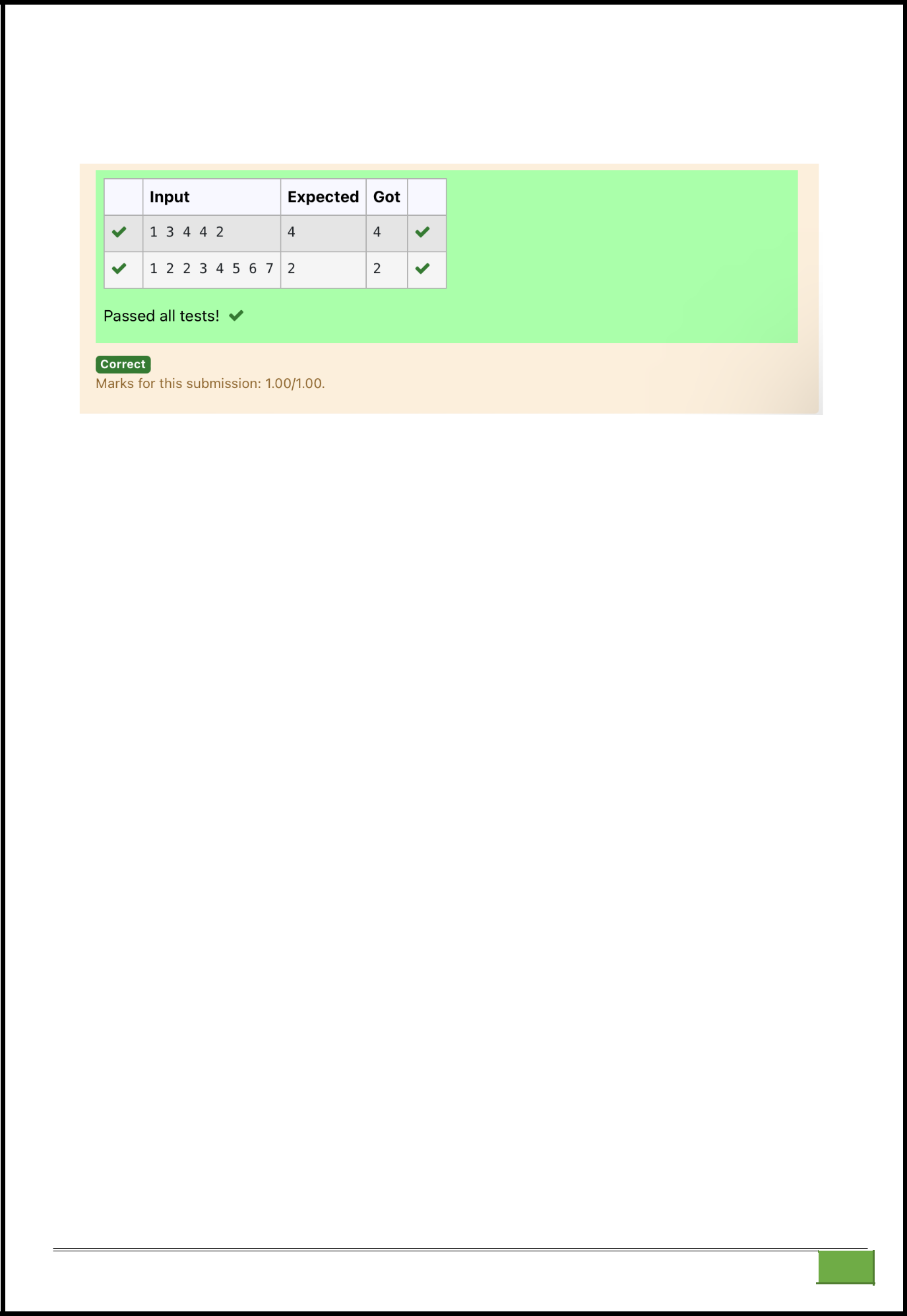
for i in nums:

if nums.count(i)==2:

b=nums.index(i)

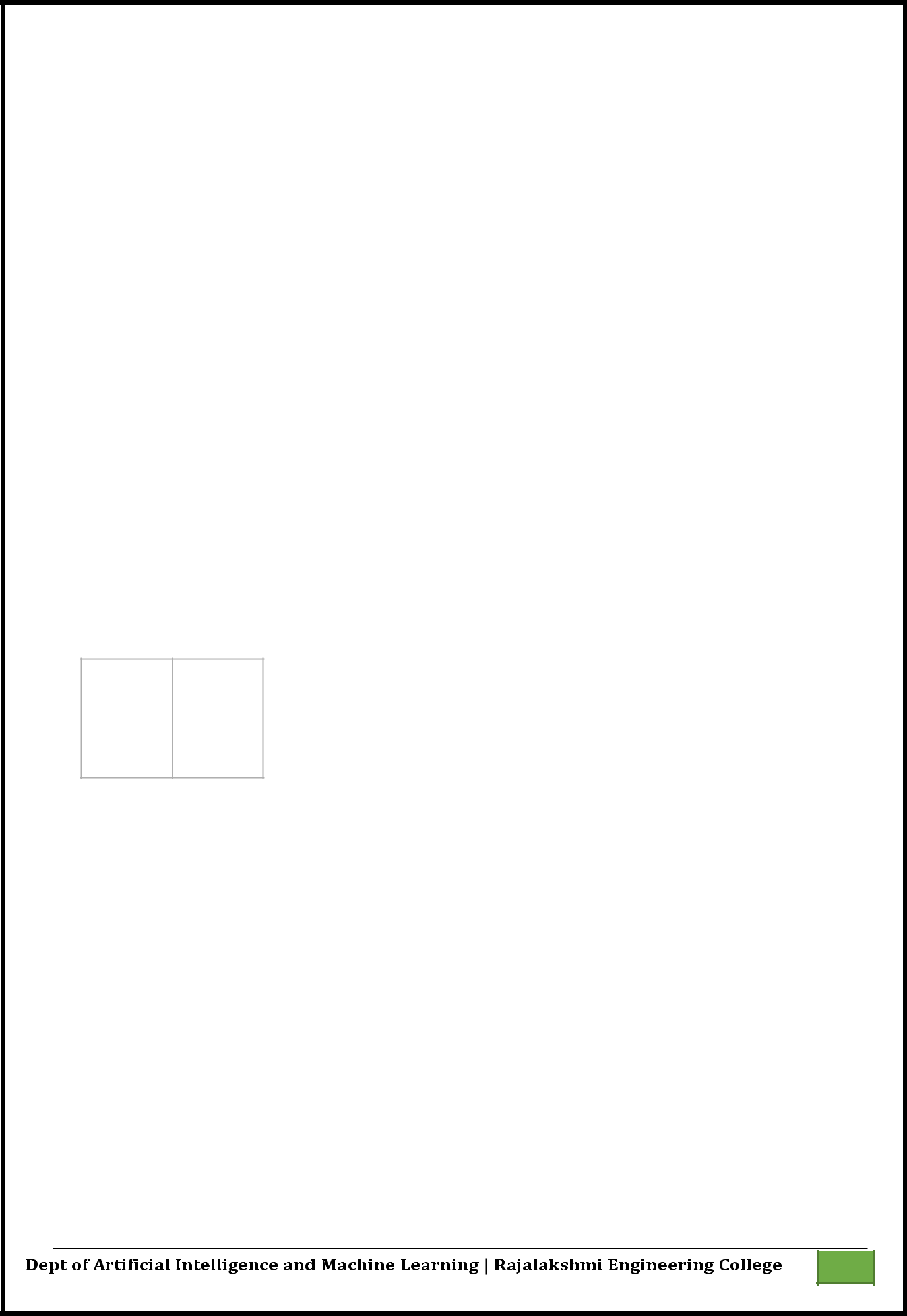
print(nums[b])

. 144

Output:

. 145

|  |  |  |  |
| --- | --- | --- | --- |
| **Ex. No.** | **:** | **8.5** | **Date:** 27/4/24 |
| **Register No.:231501044** | | | **Name: Edmond Allan A** |
|  |  |  |  |



**Remove repeated**

Write a program to eliminate the common elements in the given 2 arrays and print only the non-repeating elements and the total number of such non-repeating elements.

Input Format:

The first line contains space-separated values, denoting the size of the two arrays in [integer](https://www.rajalakshmicolleges.net/moodle/mod/quiz/view.php?id=5127) format respectively.

The next two lines contain the space-separated integer arrays to be compared.

Sample Input:

5 4

1[2865](https://www.rajalakshmicolleges.net/moodle/mod/quiz/view.php?id=5127)

26810

Sample Output:

1510

3

**For example:**

|  |  |  |
| --- | --- | --- |
| **Input** |  | **Result** |
|  |  |  |
| 5 4 |  | 1510 |
| 12865 |  | 3 |
| 26810 |  |  |
|  |  |  |

**PROGRAM**

sizes = list(map(int, input().split()))

arr1 = list(map(int, input().split()))

arr2 = list(map(int, input().split()))

set1 = set(arr1)

set2 = set(arr2)

unique1 = set1 - set2

unique2 = set2 - set1

unique\_elements = list(unique1) + list(unique2)

if unique\_elements:

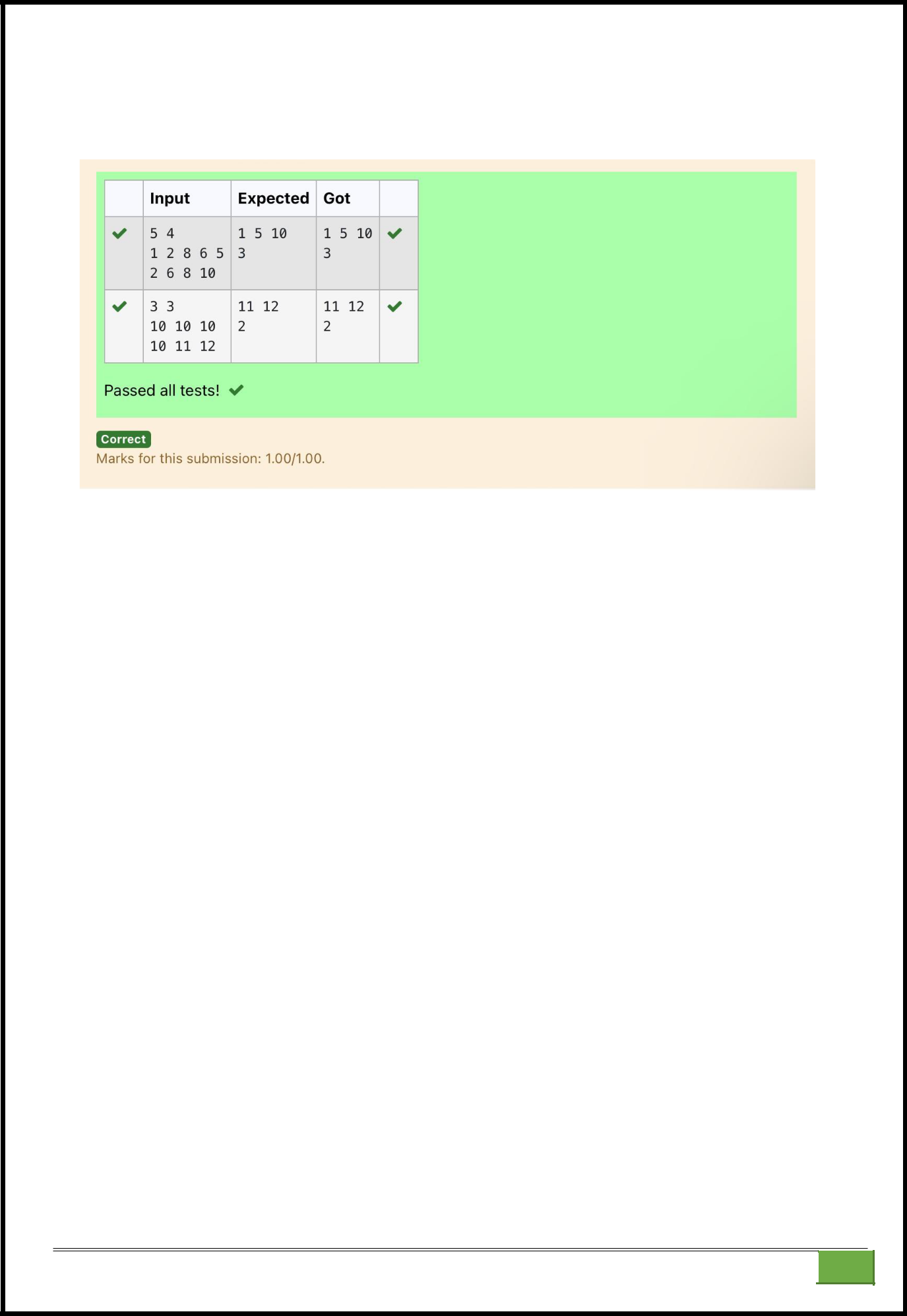
print(" ".join(map(str, unique\_elements)))

print(len(unique\_elements))

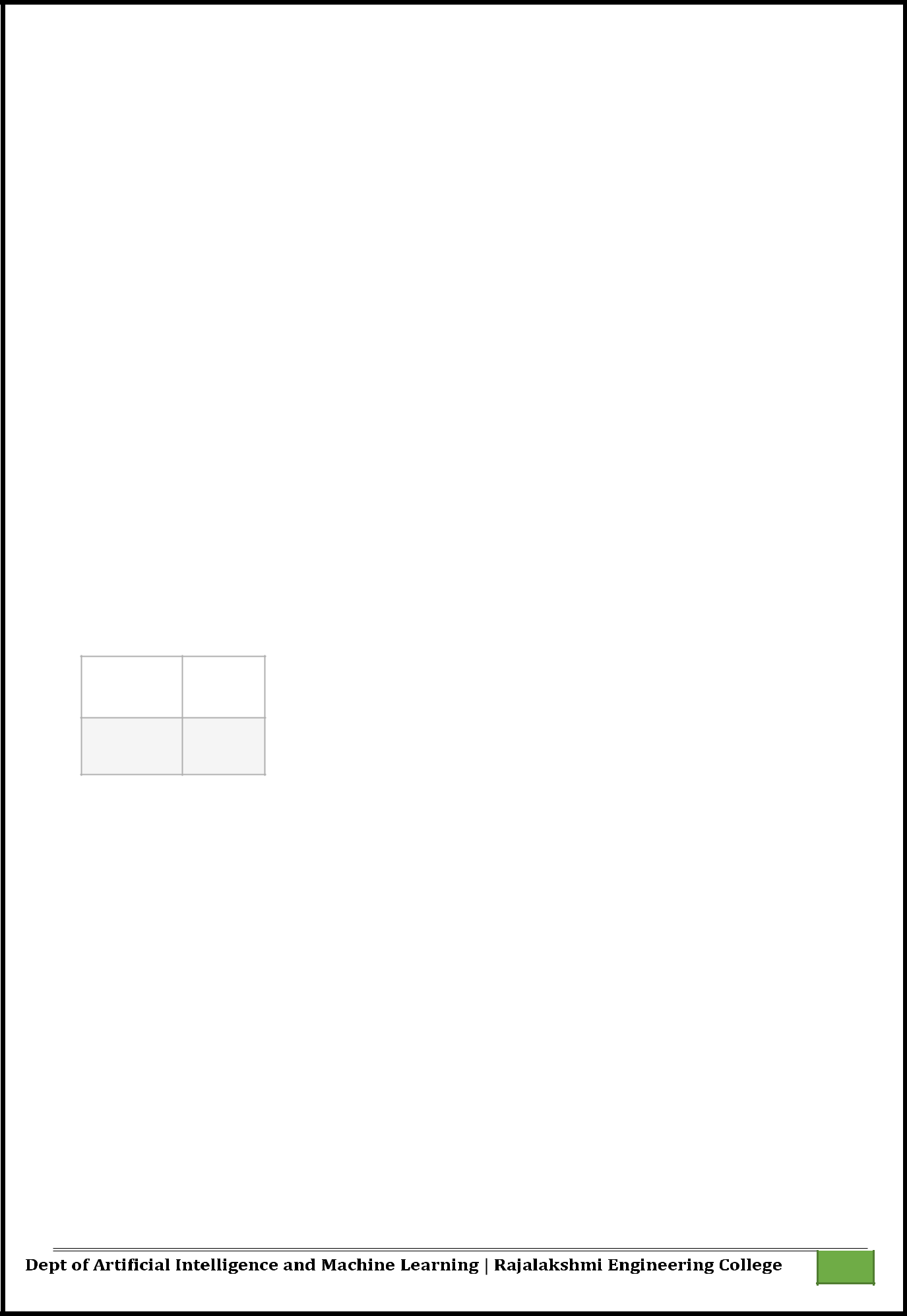
else:

print("NO SUCH ELEMENTS")

. 146

**Output:**

. 147



|  |  |  |  |
| --- | --- | --- | --- |
| **Ex. No.** | **:** | **8.6** | **Date:** 27/4/24 |
| **Register No.:231501044** | | | **Name: Edmond Allan A** |
|  |  |  |  |

**Malfunctioning Keyboard**

There is a malfunctioning keyboard where some letter keys do not work. All other keys on the keyboard work properly.

Given a string text of words separated by a single space (no leading or trailing spaces) and a string brokenLetters of all distinct letter keys that are broken, return the number of words in text you can fully type using this keyboard.

Example 1:

Input: text = "hello world", brokenLetters = "ad"

Output:

1

Explanation: We cannot type "world" because the 'd' key is broken.

**For example:**

|  |  |  |
| --- | --- | --- |
| **Input** |  | **Result** |
|  |  |  |

hello world 1

ad

**PROGRAM**

s1=str(input())

s2=str(input())

l1=(s1.lower()).split(' ')

l2=list(s2)

for i in l2:

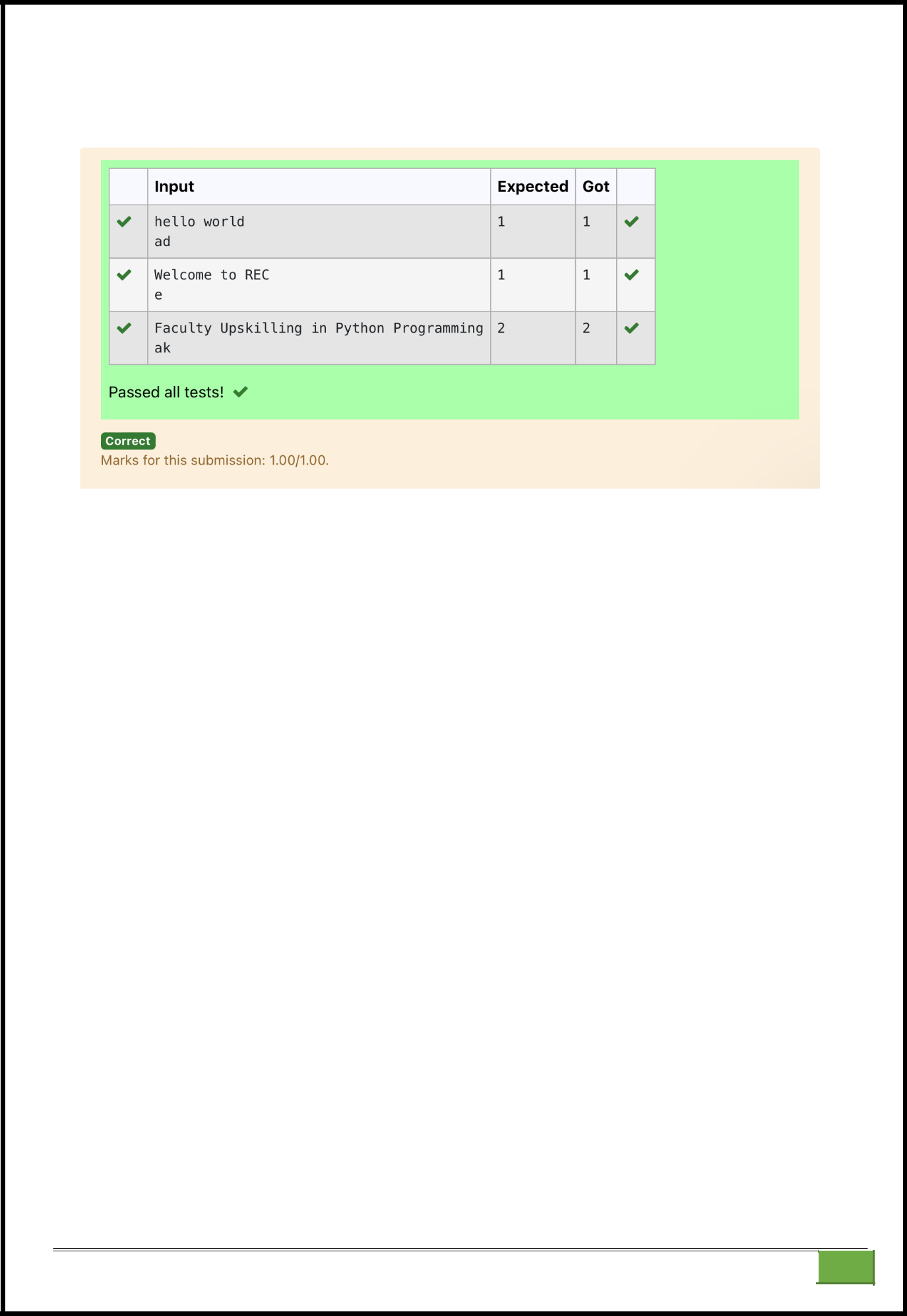
for word in l1:

if i in list(word):

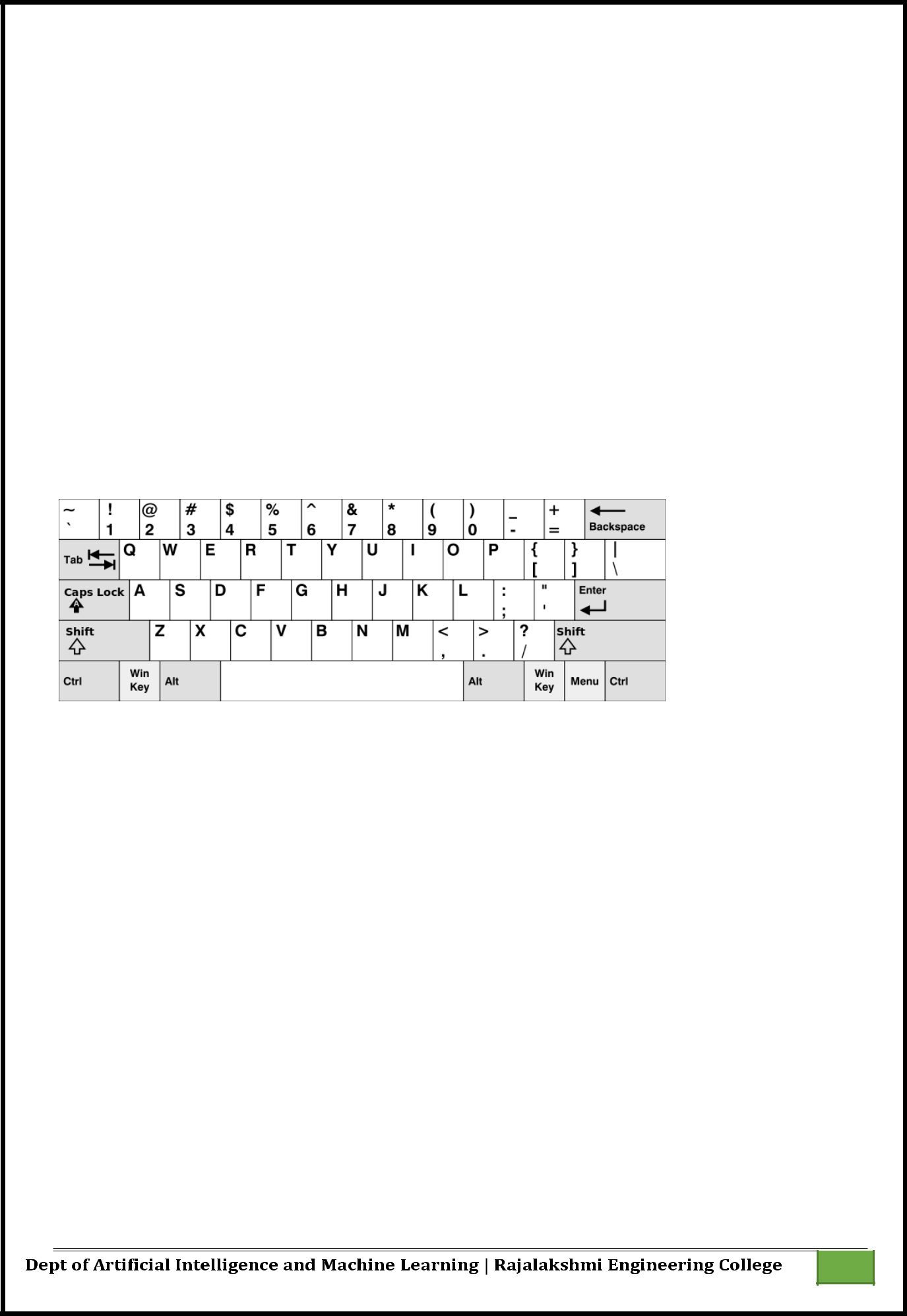
l1.remove(word)

print(len(l1))

. 148

Output:

. 149



|  |  |  |  |
| --- | --- | --- | --- |
| **Ex. No.** | **:** | **8.7** | **Date:** 27/4/24 |
| **Register No.:231501044** | | | **Name: Edmond Allan A** |
|  |  |  |  |

**American keyboard**

Given an array of strings words, return *the words that can be typed using letters of the alphabet on only one row of American keyboard like the image below*.

In the **American keyboard**:

* the first row consists of the characters "qwertyuiop",
* the second row consists of the characters "asdfghjkl", and
* the third row consists of the characters "zxcvbnm".

**Example 1:**

**Input:** words = ["Hello","Alaska","Dad","Peace"]

**Output:** ["Alaska","Dad"]

**Example 2:**

**Input:** words = ["omk"]

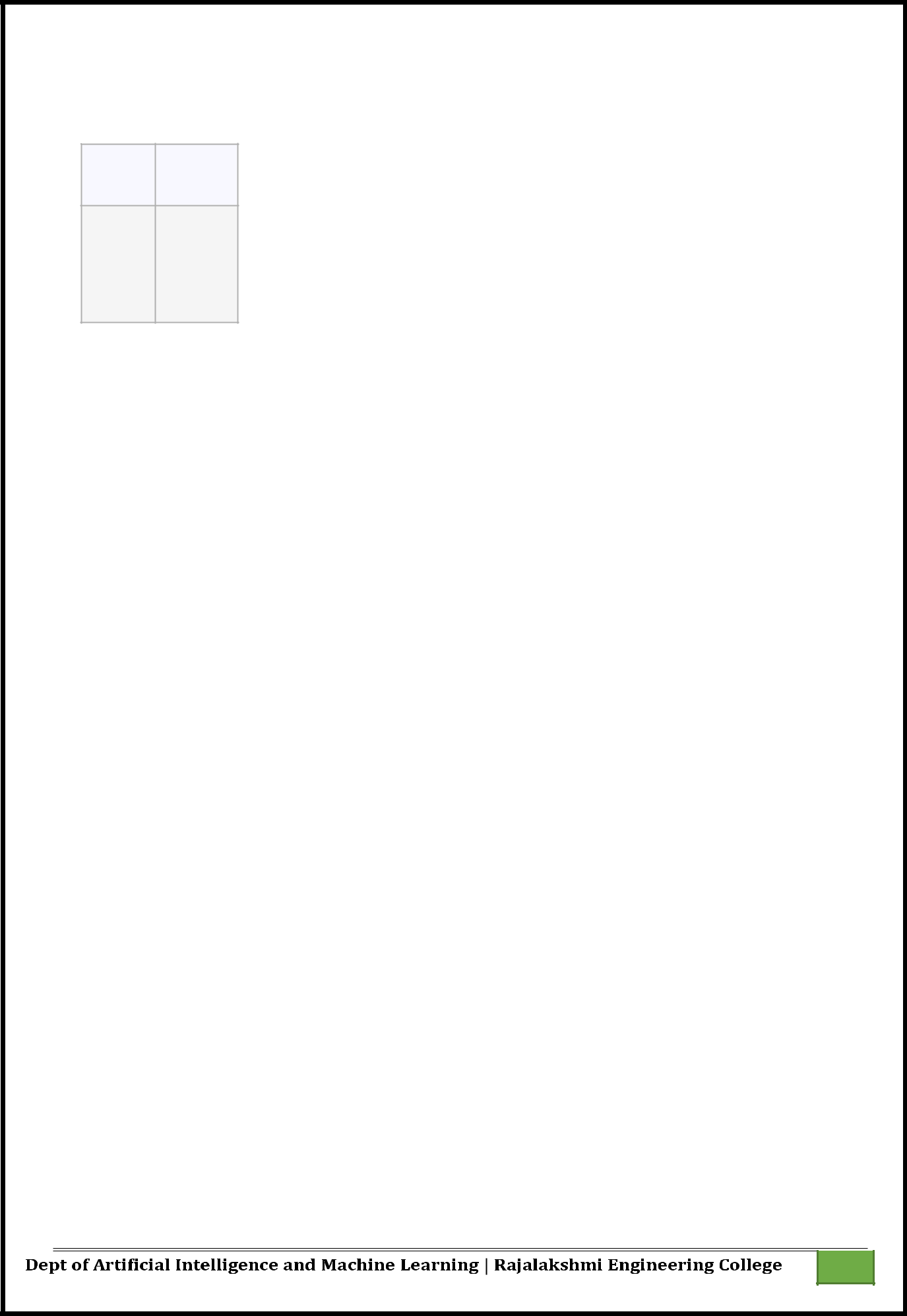
**Output:** []

**Example 3:**

**Input:** words = ["adsdf","sfd"]

**Output:** ["adsdf","sfd"]

. 150



**For example:**

**Input Result**

4

Hello

Alaska

Dad

Peace

Alaska

Dad

**PROGRAM**

n=int(input())

l=[]

for j in range(n):

l.append(str(input()))

row1 = set("qwertyuiop")

row2 = set("asdfghjkl")

row3 = set("zxcvbnm")

result = []

for word in l:

lower\_word = set(word.lower())

if lower\_word <= row1 or lower\_word <= row2 or lower\_word <= row3:

result.append(word)

if(len(result)!=0):

for i in range(len(result)):

print(result[i])

else:

print("No words")

. 151

Output:

. 152