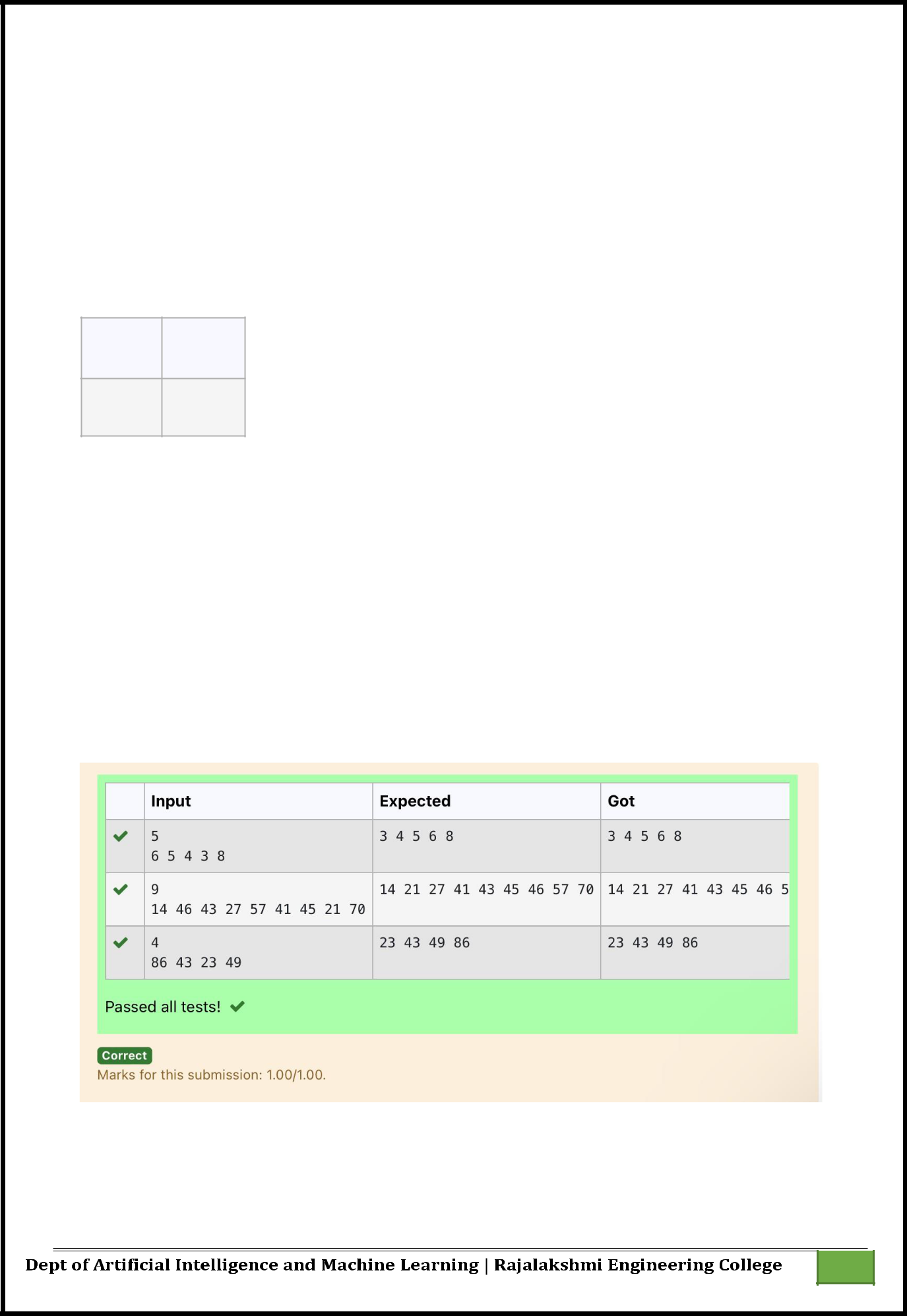


**10 - Searching & Sorting**

. 166

|  |  |  |  |
| --- | --- | --- | --- |
| **Ex. No.** | **:** | **10.1** | **Date:** 25/5/24 |
| **Register No.:231501044** | | | **Name: Edmond Allan A** |
|  |  |  |  |



**Merge Sort**

Write a Python program to sort a list of elements using the merge sort algorithm.

**For example:**

**Input Result**

1. 34568
2. 5438

**PROGRAM**

a = int(input())

b = list(input().split(" "))

b.sort()

for i in b:

print(i,end=" ")

Output:

. 167

|  |  |  |  |
| --- | --- | --- | --- |
| **Ex. No.** | **:** | **10.2** | **Date:** 25/5/24 |
| **Register No.:231501044** | | | **Name: Edmond Allan A** |
|  |  |  |  |



**Bubble Sort**

Given an listof integers, sort the array in ascending order using the *Bubble Sort* algorithm above. Once sorted, print the following three lines:

1. List is sorted in numSwaps swaps., where numSwaps is the number of swaps that took place.
2. First Element: firstElement, the *first* element in the sorted list.
3. Last Element: lastElement, the *last* element in the sorted list.

For example, given a worst-case but small array to sort: a=[6,4,1]. It took 3 swaps to sort the array. Output would be

Array is sorted in 3 swaps.

First Element: 1

Last Element: 6

**Input Format**

The first line contains an integer,n , the size of the list a .

The second line contains n, space-separated integers a[i].

**Constraints**

* 2<=n<=600
* 1<=a[i]<=2x106.

**Output Format**

You [must](http://118.185.187.137/moodle/mod/resource/view.php?id=1068) print the following three lines of output:

1. List is sorted in numSwaps swaps., where numSwaps is the number of swaps that took place.
2. First Element: firstElement, the *first* element in the sorted list[.](http://118.185.187.137/moodle/mod/resource/view.php?id=1068)
3. Last Element: lastElement, the *last* element in the sorted list.

**Sample Input 0**

3

1 2 3

**Sample Output 0**

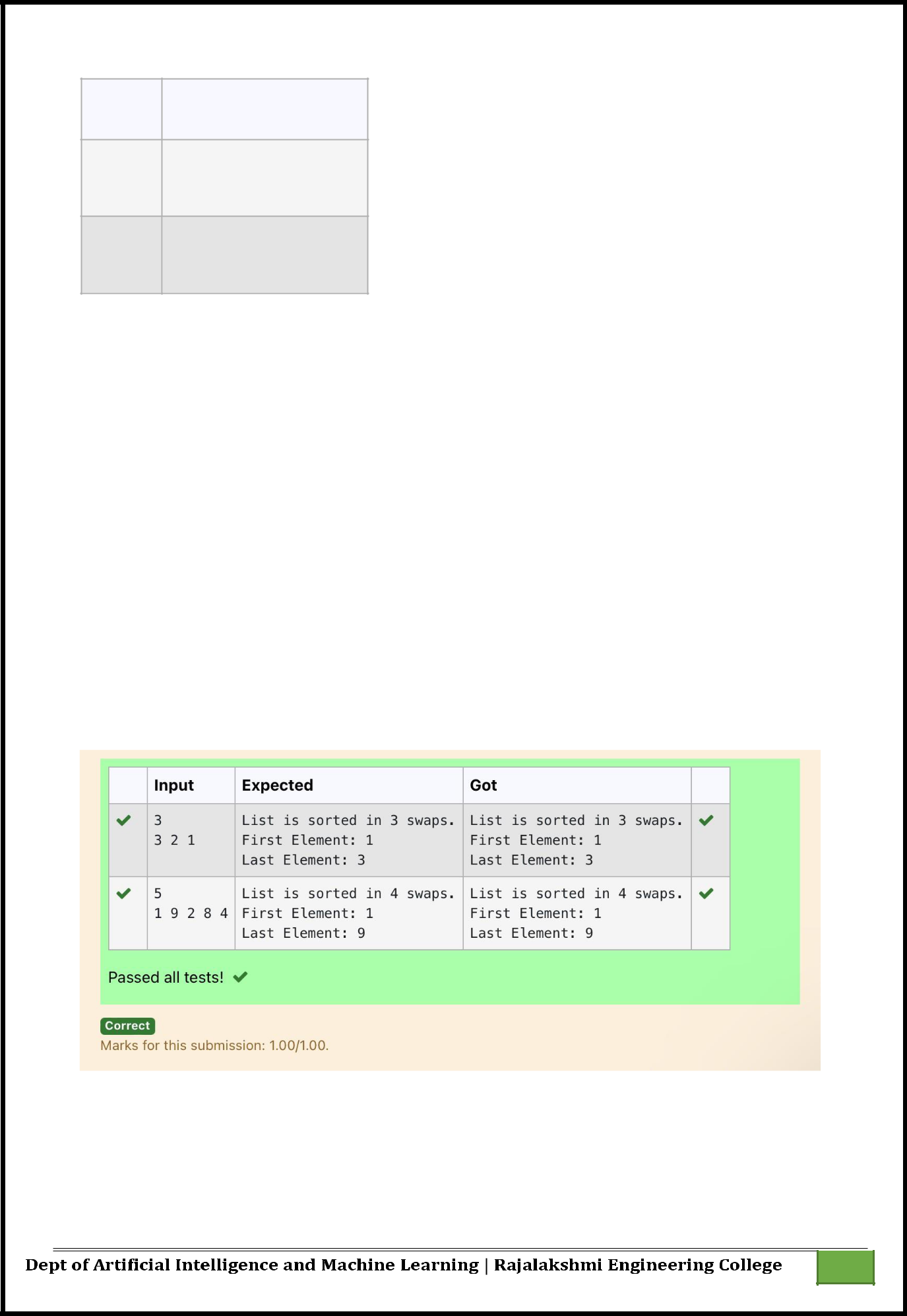
[List is](http://118.185.187.137/moodle/mod/resource/view.php?id=1068) sorted in 0 swaps.

First Element: 1

Last Element: 3

**For example:**

. 168

**Input Result**

1. List is sorted in 3 swaps.
2. 2 1First Element: 1 Last Element: 3
3. List is sorted in 4 swaps.
4. 9 2 8 4 First Element: 1 Last Element: 9

**PROGRAM**

num = 0

a = int(input())

b = input().split(" ")

c = []

for i in range(len(b)):

c.append(int(b[i]))

for j in range(len(c)):

for i in range(len(c)-1):

if c[i] > c[i+1]:

c[i], c[i+1] = c[i+1], c[i]

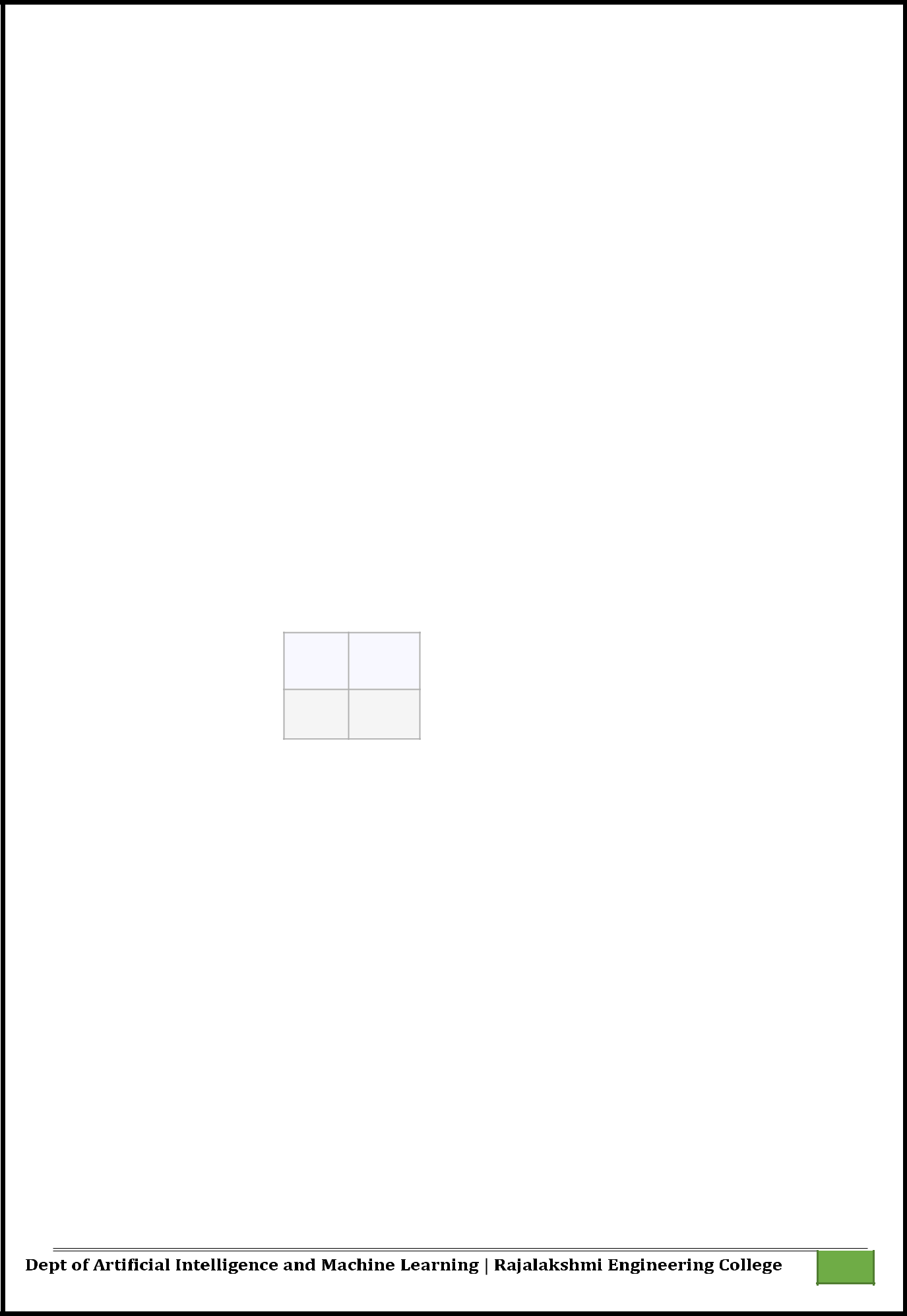
num += 1

print(f"List is sorted in {num} swaps.\nFirst Element: {c[0]}\nLast Element: {c[-1]}")

Output:

. 169

|  |  |  |  |
| --- | --- | --- | --- |
| **Ex. No.** | **:** | **10.3** | **Date:** 25/5/24 |
| **Register No.:231501044** | | | **Name: Edmond Allan A** |
|  |  |  |  |



**Peak Element**

Given an list, find peak element in it. A peak element is an element that is greater than its neighbors.

An element a[i] is a peak element if

A[i-1] <= A[i] >=a[i+1] for middle elements. [0<i<n-1]

A[i-1] <= A[i] for last element [i=n-1]

A[i]>=A[i+1] for first element [i=0]

**Input Format**

The first line contains a single integer n , the length of A .

The second line contains n space-separated integers,A[i].

**Output Format**

**Print** peak numbers separated by space.

**Sample Input**

5

891026

**Sample Output**

10 6

|  |  |
| --- | --- |
| **Input** | **Result** |
| 4 | 12 8 |
| 12368 |  |

**For example:**

**PROGRAM**

n = int(input())

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

if n == 1:

print(A[0])

else:

if A[0] >= A[1]:

print(A[0], end=" ")

for i in range(1, n - 1):

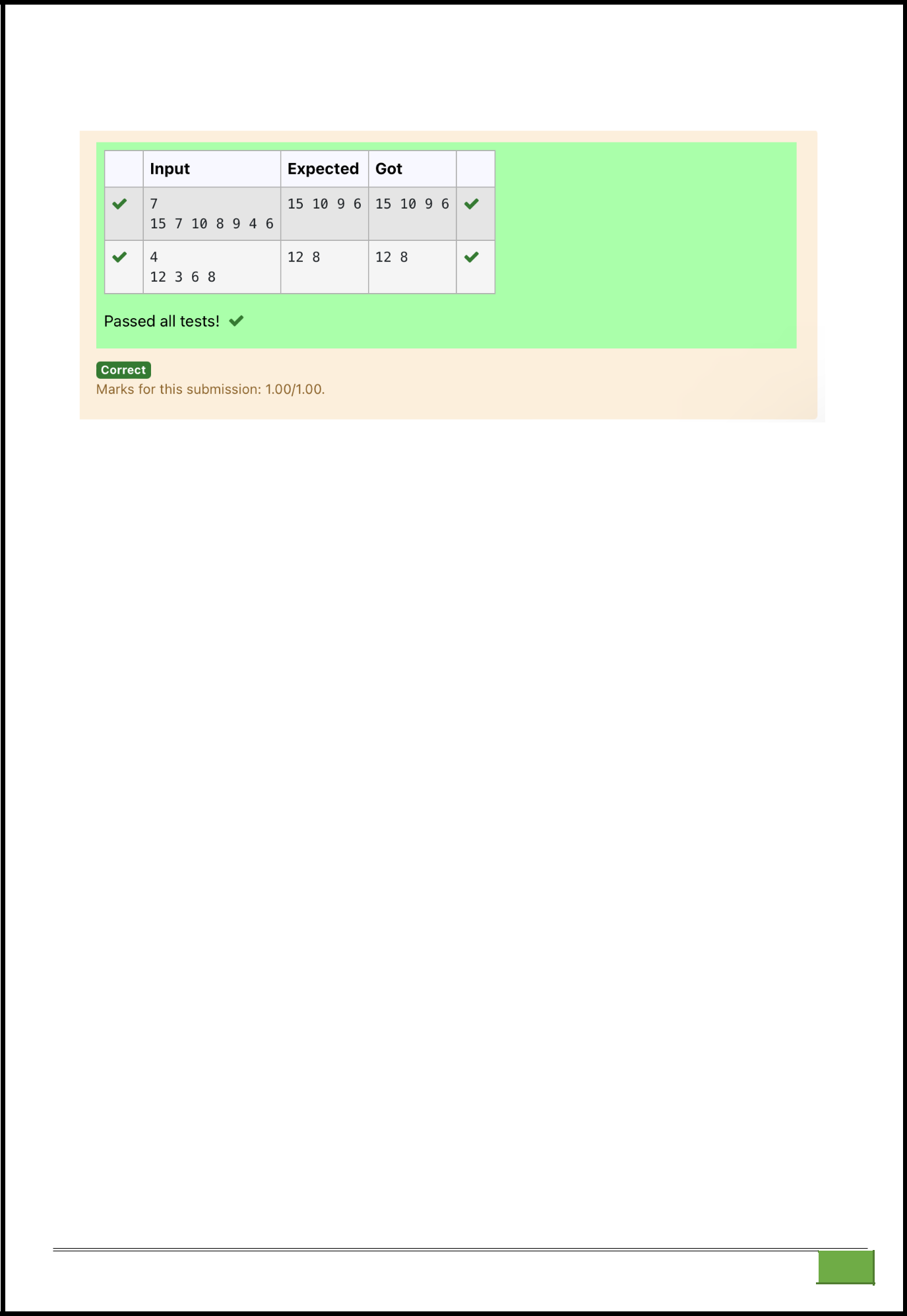
if A[i] >= A[i - 1] and A[i] >= A[i + 1]:

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

if A[n - 1] >= A[n - 2]:

print(A[n - 1])

. 170

**Output:**

. 171

|  |  |  |  |
| --- | --- | --- | --- |
| **Ex. No.** | **:** | **10.4** | **Date:** 25/5/24 |
| **Register No.:231501044** | | | **Name: Edmond Allan A** |
|  |  |  |  |



**Binary Search**

Write a Python program for binary search.

**For example:**

|  |  |
| --- | --- |
| **Input** | **Result** |
|  |  |
| 12358 | False |
| 6 |  |
|  |  |
| 3594542 | True |
| 42 |  |
|  |  |

**PROGRAM**

def binary\_search(arr, x):

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

while left <= right:

mid = (left + right) // 2

if arr[mid] == x:

return True

elif arr[mid] < x:

left = mid + 1

else:

right = mid - 1

return False

arr = list(map(int, input().split(',')))

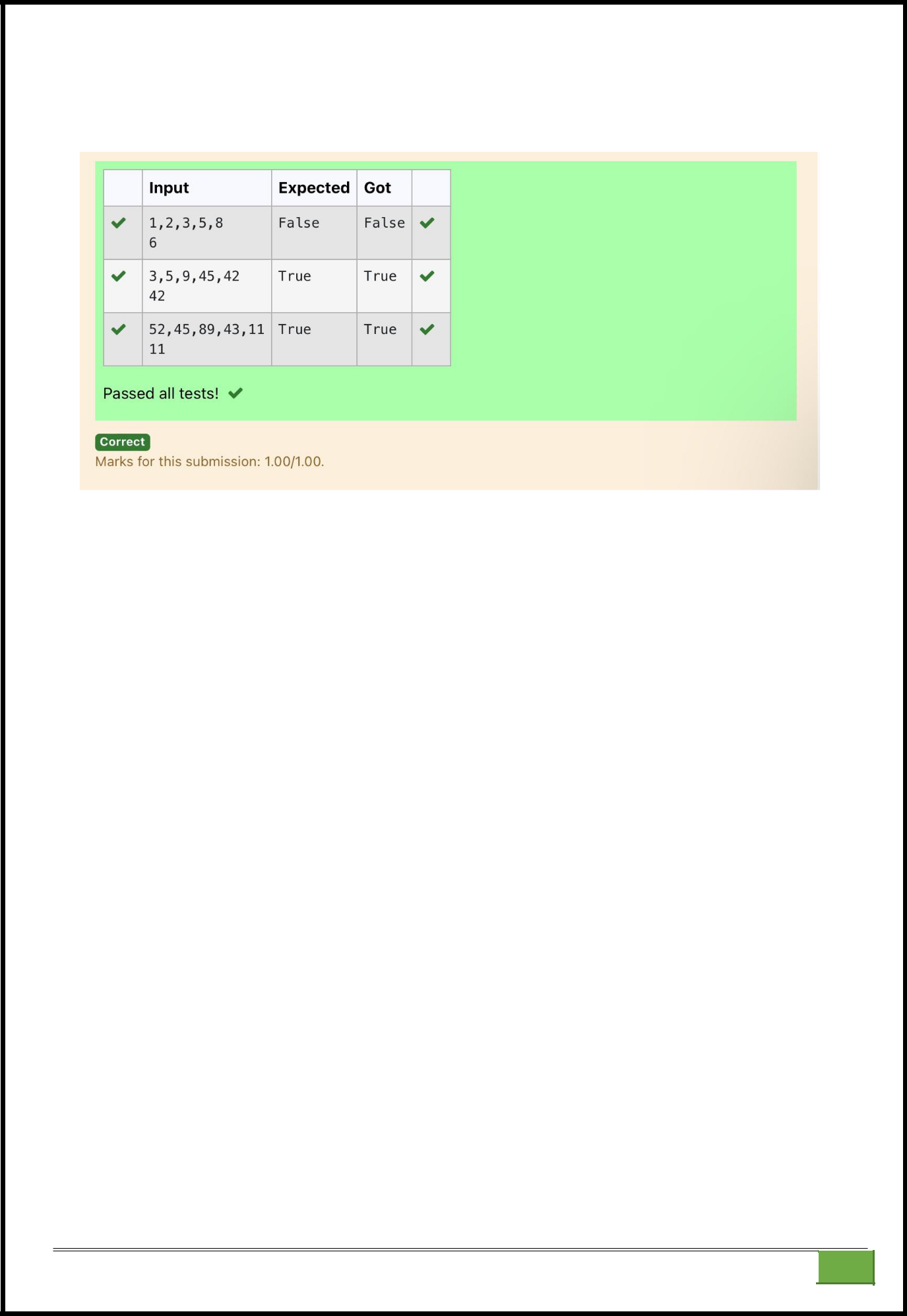
x = int(input())

arr.sort()

result = binary\_search(arr, x)

print(result)

. 172

**Output:**

. 173



|  |  |  |  |
| --- | --- | --- | --- |
| **Ex. No.** | **:** | **10.5** | **Date:** 25/5/24 |
| **Register No.:231501044** | | | **Name: Edmond Allan A** |
|  |  |  |  |

**Frequency of Elements**

To find the frequency of numbers in a list and display in sorted order.

**Constraints:**

1<=n, arr[i]<=100

**Input:**

1687949068145

**output:**

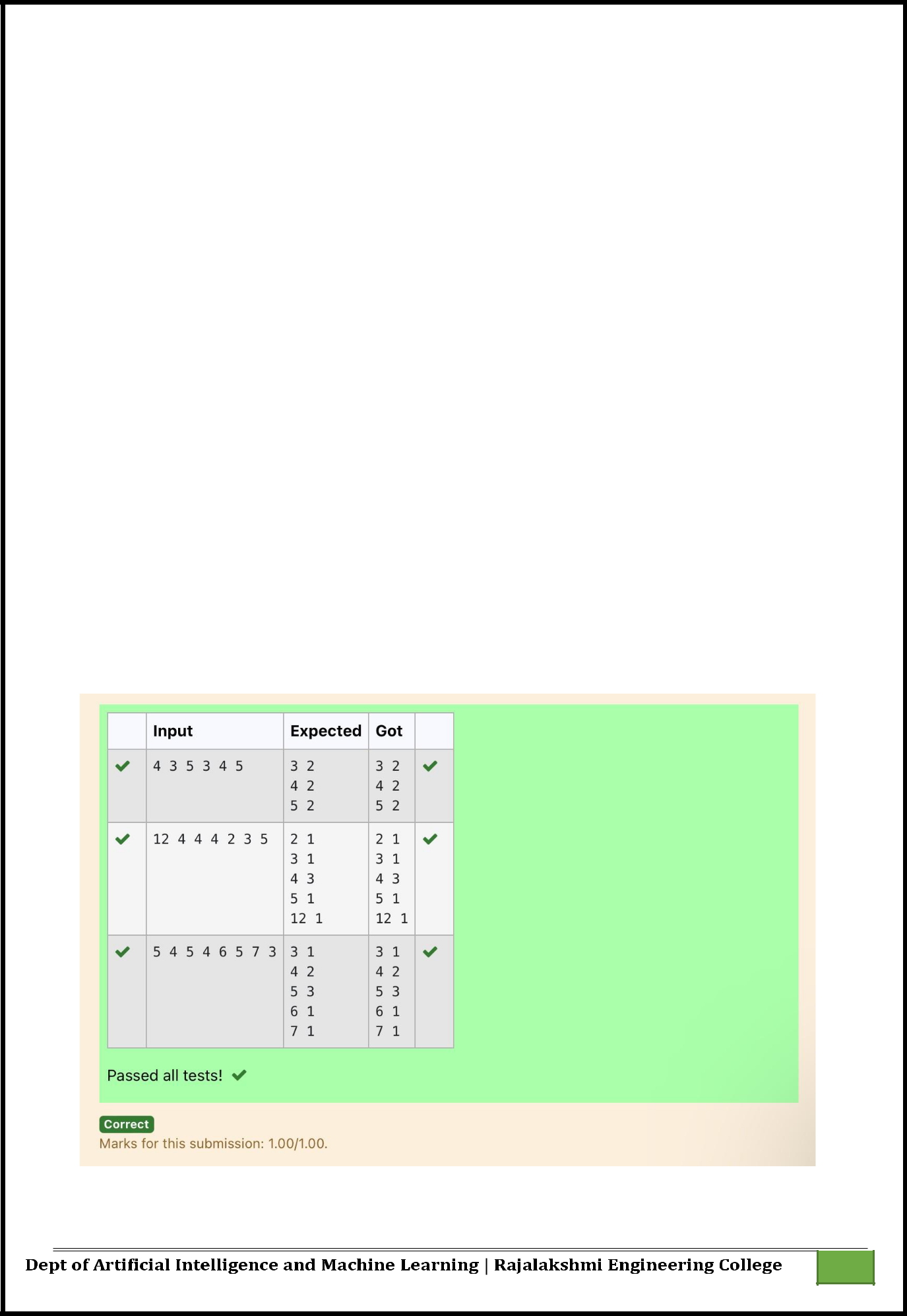
1. 2
2. 2
3. 1
4. 2
5. 1

90 1

**For example:**

|  |  |
| --- | --- |
| **Input** | **Result** |
|  |  |
| 435345 | 3 2 |
|  | 4 2 |
|  | 5 2 |
|  |  |

. 174



**PROGRAM**

numbers = input().split()

numbers = [int(num) for num in numbers]

frequency = {}

for num in numbers:

if num in frequency:

frequency[num] += 1

else:

frequency[num] = 1

sorted\_frequency = sorted(frequency.items())

for key, value in sorted\_frequency:

print(key, value)

Output:

. 175