WEEK-10-MCQ-Linear and Binary

1.In the context of searching, what is a successful search?
a. When the search algorithm finishes b.When the list contains duplicate elements
c.When the list is sorted d.When the element is found in the list

2. What is the key characteristic of binary search?
a. It can be applied only if the list is sorted
b. It works on unsorted lists
c.It always starts from the beginning of the list
d.It compares elements sequentially

3. In binary search, how is the middle element determined?
a.By starting from the first element
b.By comparing each element sequentially
c.By using a hash function
d.By dividing the list length by two

4. Given an array arr = {45,77,89,90,94,99,100} and key = 100; What are the mid values(corresponding array elements) generated in the first and second iterations?
a.90 and 99 b.90 and 100 c.89 and 94 d.94 and 99

5. ___________ search takes a sorted/ordered list and divides it in the middle.
a.Binary b.Both (1) & (3) c.Linear d.Hash

6.Which of the following is not a limitation of binary search algorithm?
a. Binary search algorithm is not efficient when the data elements more than 1500
b.Requirement of sorted array is expensive when a lot of insertion and deletions are needed
c.Must use a sorted array
d. There must be a mechanism to access middle element directly

7.In linear search, if the target element is not found in the list, what is the result?
a.The last element is returned b.An error is raised
c.The first element is returned d.The search is considered unsuccessful

8.What is the best-case time complexity of linear search?
a.O(n log n) b.O(log n) c.O(n) d.O(1)

9.Which of the following scenarios is best suited for applying binary search?
a.When the list is very small b.When the list contains duplicate elements
c.When the list is unsorted d.When the list is sorted

10.What happens when the element is found in linear search?
a.The search stops immediately
b.The search starts over from the beginning
c.The search backtracks to find duplicate elements
d.The search continues until the end of the list

11.What is the first step in binary search?
a.Sort the list
b.Divide the list into two equal parts
c.Compare the target element with the middle element in the list
d.Compare the target element with the first element in the list

12Which of the following is a conventional searching technique?
a.Linear search b.Binary search c.Hashing d.Dynamic search

13.What is the time complexity of linear search in the worst case?
a.O(log n) b.O(n) c.O(n log n) d.O(1)

14.In which situation is linear search more efficient than binary search?
a.When the list is large and unsorted b.When the list is small and sorted
c.When the list is small and unsorted d.When the list is large and sorted

15.Which of the following best describes the process of a linear search?
a.Sorting the list before searching b.Dividing the list in half repeatedly
c.Skipping every second element d.Checking each element sequentially

1. Balanced strings are those that have an equal quantity of 'L' and 'R' characters.

Given a balanced string s, split it in the maximum amount of balanced strings.

Return the maximum amount of split balanced strings.

Example 1:

Input:

RLRRLLRLRL

Output:

4

Explanation: s can be split into "RL", "RRLL", "RL", "RL", each substring contains same number of 'L' and 'R'.

Example 2:

Input:

RLLLLRRRLR

Output:

3

Explanation: s can be split into "RL", "LLLRRR", "LR", each substring contains same number of 'L' and 'R'.

Example 3:

Input:

LLLLRRRR

Output:

1

Explanation: s can be split into "LLLLRRRR".

Constraints:

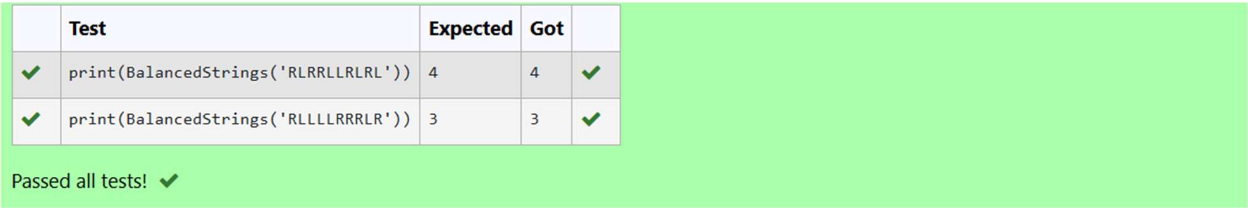
1 <= s.length <= 1000

s[i] is either 'L' or 'R'.

s is a balanced string.

Program:

```
def BalancedStrings(s,l=0,r=0,count=0):
    for i in s :
        if i=='L' :
            l+=1
        elif i=='R' :
            r+=1
        if l==r :
            count+=1
    return count
```



	Test	Expected	Got	
✓	`print(BalancedStrings('RLRRLLRLRL'))`	4	4	✓
✓	`print(BalancedStrings('RLLLLRRRLR'))`	3	3	✓

Passed all tests! ✓

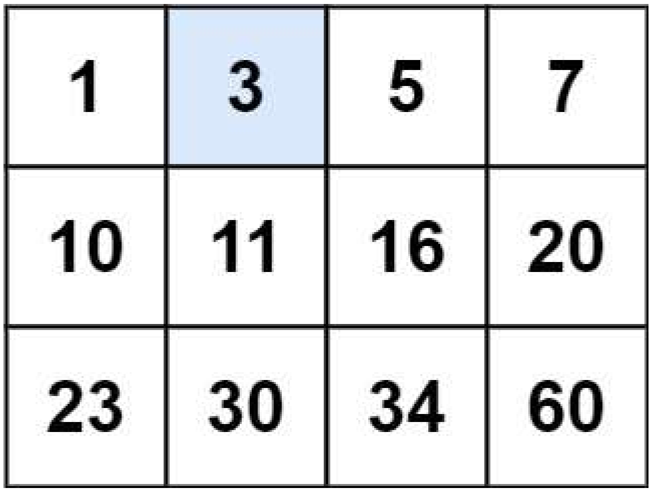
2. You are given an `m x n` integer matrix `matrix` with the following two properties:

- Each row is sorted in non-decreasing order.
- The first integer of each row is greater than the last integer of the previous row.

Given an integer `target`, return `True` *if* `target` *is in* `matrix` *or* `False` *otherwise*.

You must write a solution in `O(log(m * n))` time complexity.

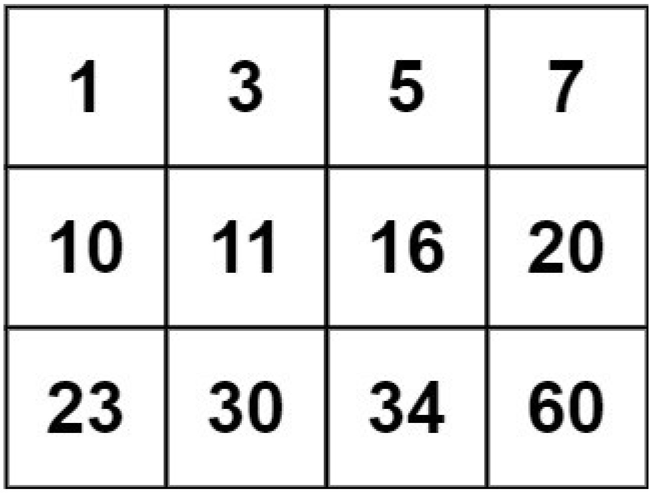
Example 1:



1	3	5	7
10	11	16	20
23	30	34	60

```
Input: matrix = [[1,3,5,7],[10,11,16,20],[23,30,34,60]], target = 3
Output: True
```

Example 2:

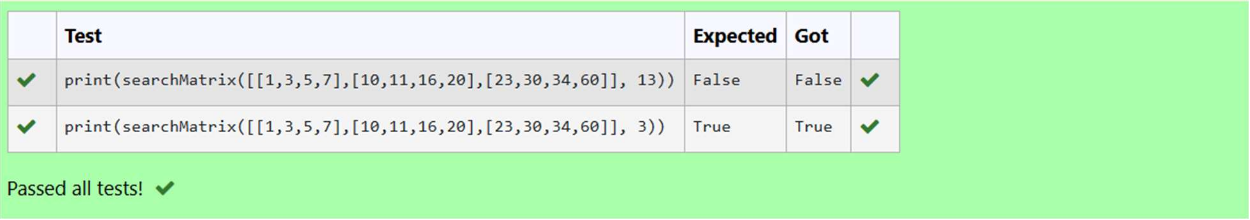


1	3	5	7
10	11	16	20
23	30	34	60

```
Input: matrix = [[1,3,5,7],[10,11,16,20],[23,30,34,60]], target = 13
Output: False
```

Program:

```
def searchMatrix(matrix:list[list[int]], target: int) -> bool:
    for i in range(len(matrix)):
        for j in range(len(matrix)):
            if matrix[i][j]==target:
                return True
    return False
```



	Test	Expected	Got	
✓	print(searchMatrix([[1,3,5,7],[10,11,16,20],[23,30,34,60]], 13))	False	False	✓
✓	print(searchMatrix([[1,3,5,7],[10,11,16,20],[23,30,34,60]], 3))	True	True	✓

Passed all tests! ✓

3. An list contains N numbers and you want to determine whether two of the numbers sum to a given number K. For example, if the input is 8, 4, 1, 6 and K is 10, the answer is yes (4 and 6). A number may be used twice.

Input Format

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

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

The third line contains integer k.

Output Format

Print Yes or No.

Sample Input

7

0 1 2 4 6 5 3

1

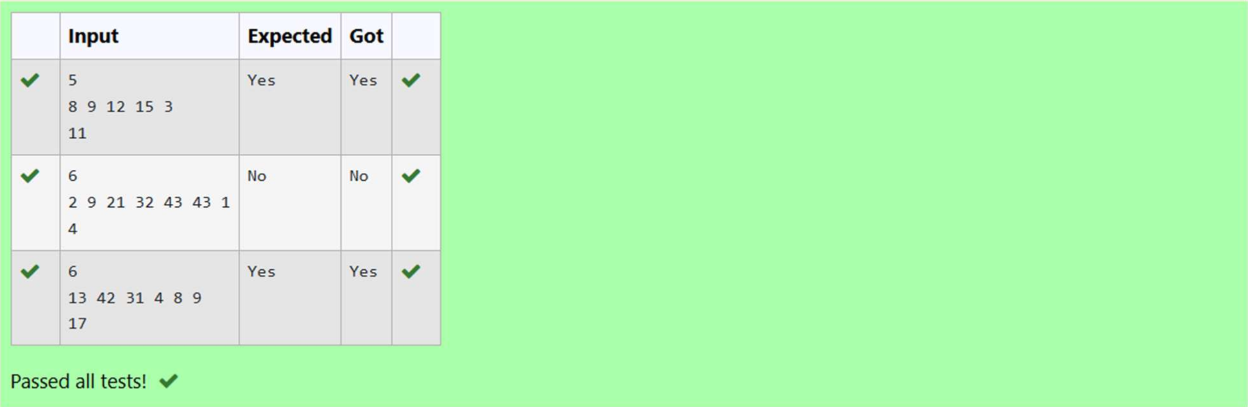
Sample Output

Yes

Program:

```
a=int(input())

p=input()

b=list(map(int,p.split()))

count=0

c=int(input())

for i in range(len(b)):

    for j in range(i+1,len(b)):

        if (b[i]+b[j])==c:

            print("Yes")

            count=1
```

```
          break
      if count==1:
        break
  if count==0:
      print("No")
```



	Input	Expected	Got	
✓	5 8 9 12 15 3 11	Yes	Yes	✓
✓	6 2 9 21 32 43 43 1 4	No	No	✓
✓	6 13 42 31 4 8 9 17	Yes	Yes	✓

Passed all tests! ✓

4. Given an array `nums` containing `n` distinct numbers in the range `[0, n]`, return *the only number in the range that is missing from the array.*

Example 1:

```
Input: nums = [3,0,1]
Output: 2
Explanation: n = 3 since there are 3 numbers, so all numbers are in the range [0,3].
2 is the missing number in the range since it does not appear in nums.
```

Example 2:

```
Input: nums = [0,1]
Output: 2
Explanation: n = 2 since there are 2 numbers, so all numbers are in the range [0,2].
2 is the missing number in the range since it does not appear in nums.
```

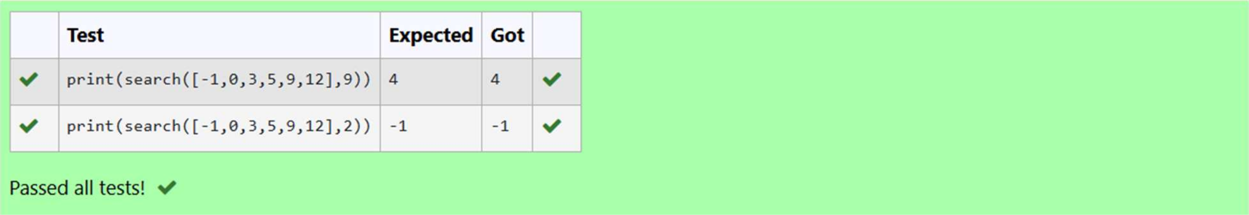
Example 3:

```
Input: nums = [9,6,4,2,3,5,7,0,1]
Output: 8
Explanation: n = 9 since there are 9 numbers, so all numbers are in the range [0,9].
8 is the missing number in the range since it does not appear in nums.
```

program:

```
    def missingNumber(n):
```

```
count=0
flag=0
p=len(n)-1
for i in range(p):
    count+=1
    if count not in n:
        flag=1
    if flag==1:
        break
if flag==1:
    return count
else:
    return n[p]+1
```



	Test	Expected	Got	
✓	`print(search([-1,0,3,5,9,12],9))`	4	4	✓
✓	`print(search([-1,0,3,5,9,12],2))`	-1	-1	✓

Passed all tests! ✓

5. Given an array of integers `nums` which is sorted in ascending order, and an integer `target`, write a function to search `target` in `nums`. If `target` exists, then return its index. Otherwise, return `-1`.

You must write an algorithm with `O(log n)` runtime complexity.

Example 1:

```
Input: nums = [-1,0,3,5,9,12], target = 9
Output: 4
Explanation: 9 exists in nums and its index is 4
```

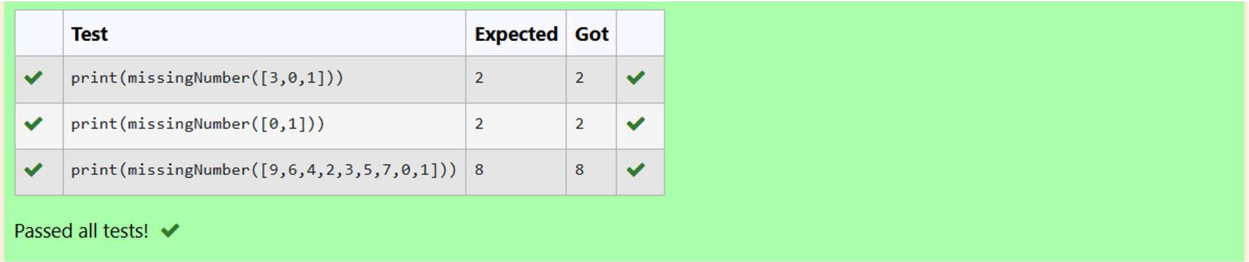
Example 2:

```
Input: nums = [-1,0,3,5,9,12], target = 2
Output: -1
Explanation: 2 does not exist in nums so return -1
```

Constraints:

- `1 <= nums.length <=` 10^4
- -10^4 `< nums[i], target <` 10^4
- All the integers in `nums` are **unique**.
- `nums` is sorted in ascending order.

```
Programdef search(num: list[int], target: int) -> int:

    count=0

    flag=0

    for i in range(len(num)):

        if num[i]==target:

            count=i

            flag=1

            break

    if flag==1:

        return count

    else:

        return -1
```



	Test	Expected	Got	
✓	`print(missingNumber([3,0,1]))`	2	2	✓
✓	`print(missingNumber([0,1]))`	2	2	✓
✓	`print(missingNumber([9,6,4,2,3,5,7,0,1]))`	8	8	✓

Passed all tests! ✓

6. Write a Python program for binary search.

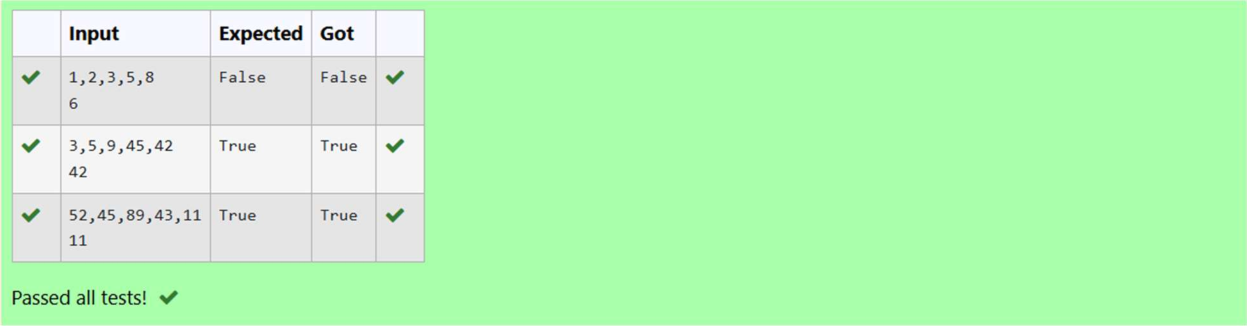
For example:

Input	Result
1,2,3,5,8 6	False
3,5,9,45,42 42	True

Program:

```
a=list(map(int,input().split(',')))
b=int(input())
c=0
flag=0
d=len(a)
a.sort()
while c<d:
    p=(c+d)//2
    if a[p]==b:
        print("True")
        flag=1
        break
    elif b<a[p]:
        d=p
    else:
        c=p+1
```

```
if flag==0:

    print("False")
```



	Input	Expected	Got	
✓	1,2,3,5,8 6	False	False	✓
✓	3,5,9,45,42 42	True	True	✓
✓	52,45,89,43,11 11	True	True	✓

Passed all tests! ✓

7. 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

8 9 10 2 6

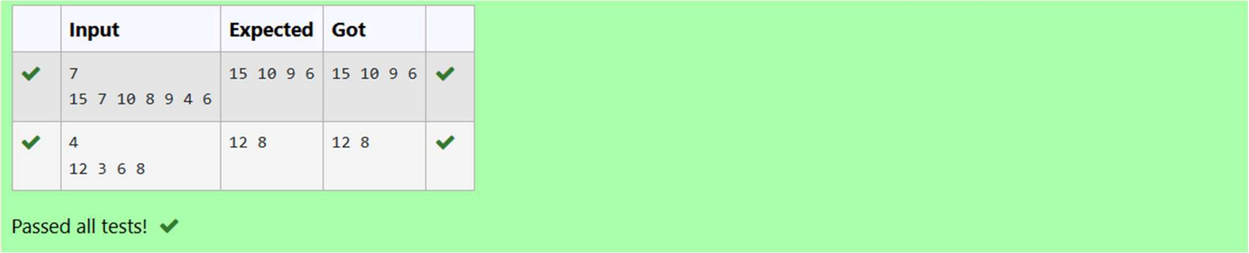
Sample Output

```
10 6
```

Program:

```
a=int(input())

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

c=[]
```

```
d=len(b)-1

if a>1:

    if b[0]>b[1]:

        c.append(b[0])

    if b[d]>b[d-1]:

        c.append(b[d])

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

    m=i-1

    n=i+1

    if b[i]>b[m] and b[i]>b[n]:

        c.append(b[i])

c.sort(reverse=True)

print(*c)
```



	Input	Expected	Got	
✓	7 15 7 10 8 9 4 6	15 10 9 6	15 10 9 6	✓
✓	4 12 3 6 8	12 8	12 8	✓

Passed all tests! ✓

8. 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
```

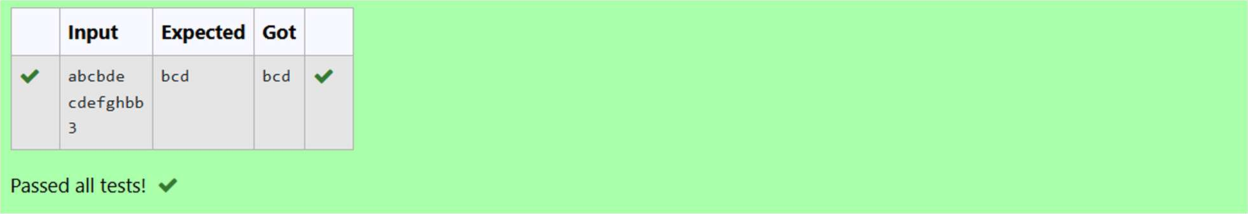
Note:

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

Program:

```
a=input()
```

```
b=input()

c=int(input())

d=""

count=0

for i in a:

    if count>=c:

        break

    if i in b and i not in d:

        d+=i

        count+=1

print(d)
```



	Input	Expected	Got	
✓	abcbde cdefghbb 3	bcd	bcd	✓

Passed all tests! ✓

9. String should contain only the words are not palindrome.

Sample Input 1

Malayalam is my mother tongue

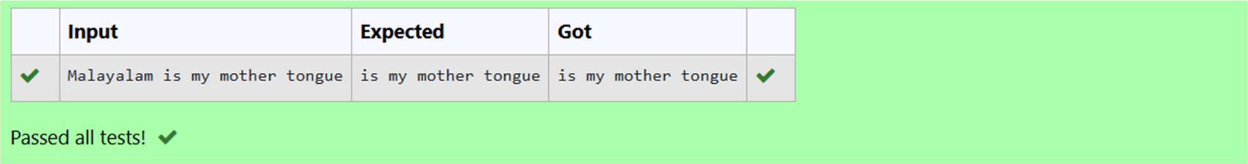
Sample Output 1

is my mother tongue

program:

```
w=input().split(' ')

u=""
```

```
for i in w:
    i=i.lower()
    if i!=i[::-1]:
        u+=i+" "
print(u)
```



	Input	Expected	Got	
✓	Malayalam is my mother tongue	is my mother tongue	is my mother tongue	✓

Passed all tests! ✓

10. 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
```

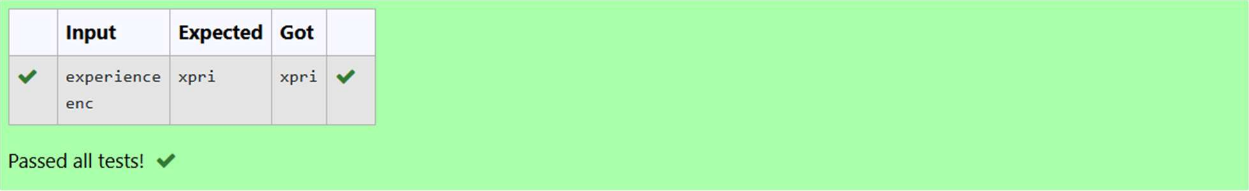
Sample Output 1

```
xpri
```

program:

```
a=input()
b=input()
c=""
for i in a:
    if i not in b:
        c+=i
```

```
print(c)
```



	Input	Expected	Got	
✓	experience enc	xpri	xpri	✓

Passed all tests! ✓