Started on	Thursday, 25 April 2024, 8:36 PM
State	Finished
Completed on	Friday, 26 April 2024, 8:12 PM
Time taken	23 hours 35 mins
Marks	5.00/5.00
Grade	**50.00** out of 50.00 (**100**%)
Name	ABINAUV R 2022-CSD-A

Question **1**

Correct

Mark 1.00 out of 1.00

Check if a set is a subset of another set.

Example:

Sample Input1:

mango apple

mango orange

mango

output1:

yes

set3 is subset of set1 and set2

```
input2:
```

```
mango orange
```

```
banana orange
```

```
grapes
```

```
output2:
```

no

Answer: (penalty regime: 0 %)

```
a = set(input())
b = set(input())
c = set(input())
if c.issubset(a):
    print("yes\nset3 is subset of set1 and set2")
else:
    print("No")
```

	Test	Input	Expected	Got	
✔	1	mango apple mango orange mango	yes set3 is subset of set1 and set2	yes set3 is subset of set1 and set2	✔
✔	2	mango orange banana orange grapes	No	No	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

Question **2**

Correct

Mark 1.00 out of 1.00

You are given an array of N integers, A1, A2, . . . , AN and an integer K. Return the of count of distinct numbers in all wi

Input :

1 2 1 3 4 3

3

Output :

2

3

3

2

Explanation

All windows of size K are

[1, 2, 1]

[2, 1, 3]

[1, 3, 4]

[3, 4, 3]

Answer: (penalty regime: 0 %)

```
def count(arr, k):
    n = len(arr)
    for i in range(n - k + 1):
        window = arr[i:i + k]
        distinct = len(set(window))
        print(distinct)
arr = list(map(int, input().split()))
k = int(input())
count(arr,k)
```

	Input	Expected	Got	
✔	1 2 1 3 4 3 3	2 3 3 2	2 3 3 2	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

Question **3**

Correct

Mark 1.00 out of 1.00

Take a complete sentence as an input and remove duplicate word in it and print (sorted order), then count all the word length greater than 3 and print.

Input

we are good are we good

Output

are good we

Count = 1

For example:

Input	Result
welcome to rec rec cse ece	cse ece rec to welcome Count = 1

Answer: (penalty regime: 0 %)

```
a = set(input().split())
print(' '.join(sorted(a)))
print("Count = {}".format(sum(1 for word in a if len(word)>3)))
```

	Input	Expected	Got	
✔	we are good are we good	are good we Count = 1	are good we Count = 1	✔
✔	welcome to rec rec cse ece	cse ece rec to welcome Count = 1	cse ece rec to welcome Count = 1	✔

Passed all tests! ✔

Question **4**

Correct

Mark 1.00 out of 1.00

Two strings, *a* and *b*, are called anagrams if they contain all the same characters in the same frequencies. For example, t
of CAT are CAT, ACT, TAC, TCA, ATC, and CTA.

Complete the function in the editor. If *a* and *b* are case-insensitive anagrams, print "Anagrams"; otherwise, print "Not A
instead.

Input Format

The first line contains a string denoting *a*.
The second line contains a string denoting *b*.

Constraints

- *1 ≤ length(a), length(b) ≤ 50*
- Strings *a* and *b* consist of English alphabetic characters.
- The comparison should NOT be case sensitive.

Output Format

Print "Anagrams" if *a* and *b* are case-insensitive anagrams of each other; otherwise, print "Not Anagrams" instead.

Sample Input 0

anagram

margana

Sample Output 0

Anagrams

Explanation 0

Character	**Frequency:** anagram	**Frequency:** margana
A or a	3	3
G or g	1	1
N or n	1	1
M or m	1	1
R or r	1	1

The two strings contain all the same letters in the same frequencies, so we print "Anagrams".

Answer: (penalty regime: 0 %)

```
a = set(input().lower())
b = set(input().lower())
if a == b:
    print("Anagrams")
else:
    print("Not Anagrams")
```

	Input	Expected	Got	
✔	madam maDaM	Anagrams	Anagrams	✔
✔	DAD DAD	Anagrams	Anagrams	✔
✔	MAN MAM	Not Anagrams	Not Anagrams	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

Question **5**

Correct

Mark 1.00 out of 1.00

Given a sorted linked list, delete all duplicates such that each element appear only *once*.

Example 1:

```
Input:
1 1 2
Output:
1 2
```

Example 2:

```
Input:
1 1 2 3 3
Output:
1 2 3
```

Answer: (penalty regime: 0 %)

```
class ListNode:
    def __init__(self, val=0, next=None):
        self.val = val
        self.next = next

def deleteDuplicates(head: ListNode) -> ListNode:
    current = head
    while current and current.next:
        if current.val == current.next.val:
            current.next = current.next.next
        else:
            current = current.next
    return head

def createLinkedListFromInput() -> ListNode:
    nums = list(map(int, input().split()))
    dummy = ListNode()
    current = dummy
    for num in nums:
        current.next = ListNode(num)
        current = current.next
    return dummy.next

def printLinkedList(head: ListNode):
    current = head
    while current:
        print(current.val, end=" ")
        current = current.next
    print()

head = createLinkedListFromInput()
result = deleteDuplicates(head)
printLinkedList(result)
```

	Test	Input	Expected	Got	
✔	1	1 1 2	1 2	1 2	✔
✔	2	1 1 2 3 3	1 2 3	1 2 3	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

◄ Week-09_MCQ

Jump to...

WI