

Solution Review: Check If List Is Sorted

This lesson will explain how to check if a list is sorted.

WE'LL COVER THE FOLLOWING ^

- Solution: Use a Loop

Solution: Use a Loop

Use a while loop that iterates over the length of the list and checks if the previous element in the list is less than the next element. If so, a variable flag is set to true. If the flag is false, a list is sorted and vice versa.

The following illustrations explain the concept of sorting a list.

Case 1: If the list is sorted

list

1	2	3	4	5
---	---	---	---	---

1 of 6

list

0	1	2	3	4
1	2	3	4	5



`list[0] < list[1]`

flag=0

2 of 6

list

0	1	2	3	4
1	2	3	4	5

↑ ↑

`list[1] < list[2]`
flag=0

3 of 6

list

0	1	2	3	4
1	2	3	4	5

↑ ↑

`list[2] < list[3]`
flag=0

4 of 6

list

0	1	2	3	4
1	2	3	4	5

↑ ↑

`list[3] < list[4]`
flag=0

5 of 6

flag=0
List is Sorted

6 of 6

—

[]

Case 2: If the list is not sorted

list

1	2	7	4	5
---	---	---	---	---

1 of 6

list

0	1	2	3	4
1	2	7	4	5

↑ ↑
`list[0] < list[1]`
flag=0

2 of 6

list

0	1	2	3	4
1	2	7	4	5

 ↑ ↑
`list[1] < list[2]`
flag=0

3 of 6

list

0	1	2	3	4
1	2	7	4	5

 ↑ ↑
`list[2] < list[3]`
flag=1

4 of 6

list

0	1	2	3	4
1	2	7	4	5

 ↑ ↑
`list[3] < list[4]`
flag=1

5 of 6

flag=1
List is UnSorted

6 of 6

—

[]

The following python code demonstates how to check if the list is sorted.

```
def isSorted(list):
    flag = 0
    i = 1
    while i < len(list):
        if(list[i] < list[i - 1]): # compare with the previous element
            flag = 1
            i += 1

    if (not flag) :
        return True
    else :
        return False
print(isSorted([1,2,3,4,5]))
print(isSorted([1,2,5,4,2]))
```



Now, let's move on to the next problem.