



Solution Review: Rearrange Positive & Negative Values

This review provides a detailed analysis of the different ways to rearrange positive and negative values in a list.

We'll cover the following



- Solution #1: Using Auxiliary Lists
 - Time Complexity
- Solution #2: Rearranging in Place
 - Time Complexity
- Solution #3: Pythonic Rearrangement
 - Time Complexity

Solution #1: Using Auxiliary Lists

```
1 def rearrange(lst):
2     neg = []
3     pos = []
4     # make a list of negative and positive numbers
5     for ele in lst:
6         if ele < 0:
7             neg.append(ele)
8         else:
9             pos.append(ele)
10    # merge two lists and return
11    return neg + pos
12
13
14 print(rearrange([10, -1, 20, 4, 5, -9, -6]))
```



15



In this solution, we use two auxiliary lists, **neg** and **pos**, to store negative and positive elements respectively. We then iterate over the entire input list and append negative elements to one list and the positive ones to the other. Finally, we simply join both auxiliary lists and return.

Time Complexity

Since the given list is only iterated over once, the time complexity of this solution is $O(n)$

Solution #2: Rearranging in Place

```
def rearrange(lst):
    leftMostPosEle = 0 # index of left most element
    # iterate the list
    for curr in range(len(lst)):
        # if negative number
        if lst[curr] < 0:
            # if not the last negative number
            if curr != leftMostPosEle:
                # swap the two
                lst[curr], lst[leftMostPosEle] = lst[leftMostPosEle], lst[curr]
            # update the last position
            leftMostPosEle += 1
    return lst

print(rearrange([10, -1, 20, 4, 5, -9, -6]))
```



In this solution, we iterate over the entire list and, if we encounter a negative element, we simply swap it with the leftmost positive element.



Time Complexity

The time complexity of this algorithm is $O(n)$ as the entire list is iterated over once, with no extra space used.

Solution #3: Pythonic Rearrangement

```
def rearrange(lst):  
    # get negative and positive list after filter and then merge  
    return [i for i in lst if i < 0] + [i for i in lst if i >= 0]  
  
print(rearrange([10, -1, 20, 4, 5, -9, -6]))
```



The solution above uses list comprehension and is much more Pythonic! It iterates over the list twice; the first time it chooses all negative numbers and second time, all the positive numbers. Finally, it joins them together and returns, all in one line.

Time Complexity

The time complexity of the solution is $O(n)$ as it is iterated over twice.


Interviewing soon? We've partnered with [Hired](#) so that companies apply to you instead of you applying to them. [See how](#) ⓘ



[← Back](#)

Challenge 9: Rearrange Positive & Ne...

Challenge 10: Rearrange Sorted List i...

 **Completed** [Report an Issue](#)