Solution Review: Find k smallest elements in a List

We'll cover the following

Solution: `removeMin()` k times#

main.py

MinHeap.py

```python
from MinHeap import MinHeap


def findKSmallest(lst, k):
    heap = MinHeap()  # Create a minHeap
    # Populate the minHeap with lst elements
    heap.buildHeap(lst)
    # Create a list of k elements such that:
    # It contains the first k elements from
    # removeMin() function
    kSmallest = [heap.removeMin() for i in range(k)]
    return kSmallest


```

```
15  lst = [9, 4, 7, 1, -2, 6, 5]
16  k = 3
17  print(findKSmallest(lst, k))
18
```

Here, we create a new heap from the given list on **line 15**. Then, we `removeMin()` from the heap k times and save the result to the list `kSmallest` using list comprehension on **line 12**. We return `kSmallest` at the end.

Time Complexity#

The time complexity of creating a heap is $O(n)$ and removing min is $O(klogn)$. So the total time complexity is $O(n + klogn)$ which is basically $O(klogn)$.

Solution #2: Using Quickselect#

You can optimize this further by calling the Quick Select algorithm on the given list k times where the input to the algorithm goes from 1 till k. We have not presented the code here because it is not relevant to heaps, but we felt that the optimal solution should be mentioned.

Time Complexity#

The *average-case* complexity of quick select is $O(n)$. So when called k times it will be in $O(nk) -> O(n)$.

Back

Challenge 2: Find k smallest elements ...

Next

Challenge 3: Find k largest elements in...

✓ Mark as Completed

Report an Issue