Sachin Singh
M24CSE033

Q. Describe an O(n)- time algorithm that, given a set S of n distinct numbers and a positive integer k≤n, determine the k numbers in S that are closest to the median of S.

To compute the k numbers near the median of a set S of n distinct numbers, it is possible to combine methods for finding medians and methods for partitioning into an O(n)-time algorithm.

1. **Find Median**: Using the median of medians algorithm, find the median of the set S in O(n) time. The median is the element at position n/2 (or (n−1)/2, if n is odd) when the set is sorted.

2. **Find the Absolute Differences**: Find the difference between each element of the set S and its median, in absolute value. Now, you will have a set D of absolute differences.

3. **Find the k-th Smallest Absolute Difference**: The quickselect algorithm is used to find the k-th smallest absolute difference from D by doing this in linear time of O(n).

4. **Partition the Set**: Having found the k-th smallest absolute difference, partition the set S in such a way that the k elements with the least absolute differences can be selected. This can easily be done using a simple linear scan in O(n) time.

5. **Returning Result**: That will give us the k elements closest to the median, the ones that have the smallest absolute differences to the median.

Example

```
int S[] = {12, 3, 17, 13, 26, 9, 5, 8, 15, 10, 7};
int n = sizeof(S) / sizeof(S[0]);
int k = 4;  // Find 4 numbers closest to the median

findKClosestToMedian(S, n, k);
```

Output

```
********** Sachin Singh **********
********** M24CSE033 **********
--------------------------------------------------

Median: 10
The 4 numbers closest to the median are:
12 9 8 10


...Program finished with exit code 0
Press ENTER to exit console.
```