167) To Implement a function median_of_medians(arr, k) that takes an unsorted array arr and an integer k, and returns the k-th smallest element in the array.

arr = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10] k = 6

arr = [23, 17, 31, 44, 55, 21, 20, 18, 19, 27] k = 5

Output: An integer representing the k-th smallest element in the array.

AIM: That takes an unsorted array arr and an integer k, and returns the k-th smallest element in the array.

PROGRAM:

```
def find_median(arr):
    arr.sort()
    return arr[len(arr)//2]

def median_of_medians(arr, k):
    if len(arr) == 1:
        return arr[0]

    # Divide arr into groups of size 5
    groups = [arr[i:i+5] for i in range(0, len(arr), 5)]

    # Find median of each group
    medians = [find_median(group) for group in groups]

    # Find the median of medians recursively
    pivot = median_of_medians(medians, len(medians)//2)

    # Partition the array around the pivot
    left = [x for x in arr if x < pivot]
    right = [x for x in arr if x > pivot]
    equal = [x for x in arr if x == pivot]

    if k <= len(left):
        return median_of_medians(left, k)
    elif k > len(left) + len(equal):
        return median_of_medians(right, k - len(left) - len(equal))
    else:
        return pivot

# Examples
arr1 = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
k1 = 6
print(median_of_medians(arr1, k1))  # Output: 6
```

OUTPUT: 

TIME COMPLEXITY : O(N)