161)Implement the Quick Sort algorithm in a programming language of your choice and test it on the array 19,72,35,46,58,91,22,31. Choose the middle element as the pivot and partition the array accordingly. Show the array after this partition. Recursively apply Quick Sort on the sub-arrays formed. Display the array after each recursive call until the entire array is sorted. Execute your code and show the sorted array.
Input : N= 8, a[] = {19,72,35,46,58,91,22,31}
Output : 19,22,31,35,46,58,72,91
Test Cases :
Input : N= 8, a[] = {31,23,35,27,11,21,15,28}
Output : 11,15,21,23,27,28,31,35
Test Cases :
Input : N= 10, a[] = {22,34,25,36,43,67, 52,13,65,17}
Output : 13,17,22,25,34,36,43,52,65,67

AIM: Show the array after this partition. Recursively apply Quick Sort on the sub-arrays formed. Display the array after each recursive call until the entire array is sorted. Execute your code and show the sorted array.

PROGRAM :

```
def quick_sort(arr, start, end):
    if start < end:
        p = partition(arr, start, end)
        print(f"Array after partitioning with pivot {arr[p]}: {arr}")
        quick_sort(arr, start, p - 1)
        quick_sort(arr, p + 1, end)


def partition(arr, start, end):
    mid = (start + end) // 2
    pivot = arr[mid]
    arr[mid], arr[end] = arr[end], arr[mid]  # Move pivot to end for simplicity
    i = start
    for j in range(start, end):
        if arr[j] < pivot:
            arr[i], arr[j] = arr[j], arr[i]
            i += 1
```

```
    arr[i], arr[end] = arr[end], arr[i]  # Move pivot to its correct place
    return i

def quick_sort_wrapper(arr):
    quick_sort(arr, 0, len(arr) - 1)
    print(f"Final sorted array: {arr}")

arr = [19, 72, 35, 46, 58, 91, 22, 31]
print(f"Original array: {arr}")
quick_sort_wrapper(arr)

test_cases = [
    [31, 23, 35, 27, 11, 21, 15, 28],
    [22, 34, 25, 36, 43, 67, 52, 13, 65, 17]
]

for test in test_cases:
    print(f"\nOriginal array: {test}")
    quick_sort_wrapper(test)
```

INPUT:

```
Original array: [19, 72, 35, 46, 58, 91, 22, 31]
```

OUTPUT:

```
Array after partitioning with pivot 72: [19, 22, 31, 35, 46, 58, 72, 91]
Final sorted array: [19, 22, 31, 35, 46, 58, 72, 91]
```

TIME COMPLEXITY : O(N^2)