**Experiment No: 3**

**QUICK SORT**

**Aim:** To implement quick sort using divide and conquer.

**Theory:**

**Quick Sort:**

Quick sort is a highly efficient sorting algorithm and is based on partitioning of array of data into smaller arrays. A large array is partitioned into two arrays one of which holds values smaller than the specified value, say pivot, based on which the partition is made and another array holds values greater than the pivot value.

Quicksort partitions an array and then calls itself recursively twice to sort the two resulting subarrays. This algorithm is quite efficient for large-sized data sets as its average and worst-case complexity are O(n2), respectively.

**Algorithm:**

Consider an Array( A ) with N elements having LB as lowerbound and UB as upperbound.

**ALGORITHM : QUICK\_SORT(A, LB, UB )**

Step 1 If ( LB < UB ) then

Loc = PARTITION (A, LB, UB )

QUICK\_SORT(A, LB, Loc-1 )

QUICK\_SORT(A, Loc+1, UB )

Else

// Print sorted Array

Step 2 End

**ALGORITHM PARTITION(A, LB, UB)**

{

pivot = start;

start= LB;

end= UB;

while (start < end)

{

while (A[start] <= A[pivot] and start<end)

start++;

while (A[end] > A[pivot])

end--;

if (start < end)

{

Swap(A[start], A[end]) ;

}

}

Swap(A[LB], A[end]);

return end;

}

**PROGRAM:**

Conclusion: Quick sort using divide and conquer was studied and implemented successfully.