Course Name: DAA Lab **Course Code: 21ITH-311/21CSH-311**

Experiment 2.1

Aim: Sort a given set of elements using the Quick sort method and determine the time required to sort the elements. Repeat the experiment for different values of n, the number of elements in the list to be sorted. The elements can be read from a file or can be generated using the random number generator.

Objectives: To understand quick sort.

Input/Apparatus Used: Quick sort concept is used.

Procedure/Algorithm:

Quick Sort is a Divide and Conquer algorithm. It picks an element as pivot and partitions the given array around the picked pivot. There are many different versions of quick Sort that pick pivot in different ways.

● Always pick first element as pivot.

● Always pick last element as pivot (implemented below)

● Pick a random element as pivot.

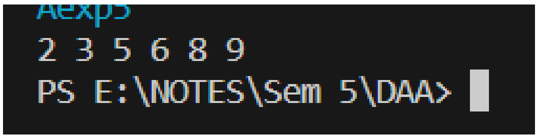
● Pick median as pivot.

The key process in quick Sort is partition (). Target of partition() is, given an array and an element x of array as pivot, put x at its correct position in sorted array and put all smaller elements (smaller than x) before x, and put all greater elements (greater than x) after x.

Sample Code:

```
public class DAAexp5 {
   public static void printArr(int arr[]){
      for(int i=0; i<arr.length; i++){
          System.out.print(arr[i]+" ");
      }
      System.out.println();
   }

   public static void quickSort(int arr[], int si, int ei){
      if(si >= ei){
          return;
      }

      int pIdx = partition(arr, si, ei);
      quickSort(arr, si, pIdx-1);
      quickSort(arr, pIdx+1, ei);
   }

   public static int partition(int arr[], int si, int ei){
      int pivot = arr[ei];
      int i = si-1;

      for(int j=si; j<ei; j++){
          if(arr[j] <= pivot){
              i++;
              //swap
              int temp =arr[j];
              arr[j] = arr[i];
              arr[i] = temp;
          }
      }
      i++;
      int temp = pivot;
      arr[ei] = arr[i];
      arr[i] = temp;
      return i;
   }
   public static void main(String[] args) {
      int arr[] = {6, 3, 9, 8, 2, 5};
      quickSort(arr, 0, arr.length-1);
      printArr(arr);
   }
}
```

Observations/Outcome :



```
Aexp5
2 3 5 6 8 9
PS E:\NOTES\Sem 5\DAA>
```

Time Complexity:

Worst Case: O(n^2)
Average Case: O(nlogn)
Best Case: O(nlogn)