Program 14 **To Implement Quick Sort**

Algorithm

Consider a linear array having N elements and it is sorted using Quick sort technique. We will consider high and low from array and then the array will be divided in sub lists accordingly and then will be sorted.

Step 1:-Call quick sort function

Step 2:- Array, LOW, HIGH will be passed in function.

[IN FUNCTION] make another function of PARTITION and two other functions of QUICK\_SORT.

Step 3:-print the sorted array

Step 4:-End

PARTITION (a[],LOW,HIGH)

Step 1:-[initialize] PIVOT, LOW, HIGH

Step 2:-Set PIVOT = a[LOW] , i = LOW, j = HIGH+1

Step 3:-repeat step 4 till i < j

Step 5:-repeat step 6 till a[LOW] < PIVOT

Step 6:-i++

[End of Step 3]

Step 7:- repeat step 8 till a[HIGH] > PIVOT

Step 8:- j--

[End of Step 5]

Step 9:-IF i < j

SWAP(a[LOW],a[HIGH])

[End of IF structure]

Step 10:- SWAP (a[LOW],a[HIGH])

Step 11:-SWAP(a[HIGH],PIVOT)

Step 12:-End

***SOURCE CODE***

#include<stdio.h>

#include<stdlib.h>

int quick\_sort(int[], int ,int);

int partition(int[], int, int);

void main()

{

int a[1000],i,n;

system("COLOR F0");

printf("Enter the range of array=");

scanf("%d",&n);

printf("\nEnter the array\n");

for(i=0;i<n;i++)

{

scanf("%d",&a[i]);

}

quick\_sort(a , 0 , n-1);

printf("\n");

for(i=0;i<n;i++)

{

printf("%d\n",a[i]);

}

}

int quick\_sort(int a[],int l,int u)

{

int j;

if(l<u)

{

j=partition(a , l ,u);

quick\_sort(a, l , j-1);

quick\_sort(a, j+1 , l);

}

}

int partition(int a[] , int l , int u)

{

int pivot i , j , temp;

pivot = a[l];

i=l;

j=u+1;

do

{

do

{

i++;

}while(a[i]<pivot&&i<=u);

do

{

j--;

}while(a[j]>pivot);

if(i<j)

{

temp=a[i];

a[i]=a[j];

a[j]=temp;

}

}while(i<j);

a[l]=a[j];

a[j]=pivot;

return (j);

}

