S.No: 22	Exp. Name: ***Write a C program to sort given elements using quick sort***	Date:2023-04-06

ID: 224G1A0516

Aim:

Write a program to sort (Ascending order) the given elements using quick sort technique.

Note: Pick the first element as pivot. You will not be awarded marks if you do not follow this instruction.

At the time of execution, the program should print the message on the console as:

```
Enter array size :
```

For example, if the user gives the input as:

```
Enter array size : 5
```

Next, the program should print the following message on the console as:

```
Enter 5 elements :
```

if the user gives the input as:

```
Enter 5 elements : 34 67 12 45 22
```

then the program should print the result as:

```
Before sorting the elements are : 34 67 12 45 22
After sorting the elements are : 12 22 34 45 67
```

Note: Do use the **printf()** function with a newline character **(\n).**

Source Code:

quicksortmain.c

```
#include<stdio.h>
void read(int [],int);
void display(int [],int);
void main()
{
   int arr[15],i,n;
   printf("Enter array size : ");
   scanf("%d",&n);
   printf("Enter %d elements : ",n);
   for(i=0;i<n;i++)
   {
      scanf("%d",&arr[i]);
   }
   printf("Before sorting the elements are : ");
   display(arr,n);
   quickSort(arr,0,n-1);
   printf("After sorting the elements are : ");
   display(arr,n);
}
int i,j,temp,pivolt,left,right;
void display(int a[15],int n)
{
```

```
   for(i=0;i<n;i++)
   printf("%d ",a[i]);
   printf("\n");
}
int partition(int arr[15],int lb,int ub)
{

}
void quickSort(int a[15],int low,int high)
{
   left=low;
   right=high;
   pivolt=a[(low+high)/2];
   do
   {
      while(a[left]<pivolt)
      left++;
      while(a[right]>pivolt)
      right--;
      if(left<=right)
      {
         temp=a[left];
         a[left]=a[right];
         a[right]=temp;
         right--;
         left++;
      }
   }
   while(left<=right);
   if(low<right)
   quickSort(a,low,right);
   if(left<high)
   quickSort(a,left,high);
}
```

quicksortfunctions.c

```
#include<stdio.h>
void read(int [],int);
void display(int [],int);
void main()
{
   int arr[15],i,n;
   printf("Enter array size : ");
   scanf("%d",&n);
   printf("Enter %d elements : ",n);
   for(i=0;i<n;i++)
   {
      scanf("%d",&arr[i]);
   }
   printf("Before sorting the elements are : ");
   display(arr,n);
   quickSort(arr,0,n-1);
   printf("After sorting the elements are : ");
   display(arr,n);
```

```
}
int i,j,temp,pivolt,left,right;
void display(int a[15],int n)
{
   for(i=0;i<n;i++)
   printf("%d ",a[i]);
   printf("\n");
}
int partition(int arr[15],int lb,int ub)
{

}
void quickSort(int a[15],int low,int high)
{
   left=low;
   right=high;
   pivolt=a[(low+high)/2];
   do
   {
      while(a[left]<pivolt)
      left++;
      while(a[right]>pivolt)
      right--;
      if(left<=right)
      {
         temp=a[left];
         a[left]=a[right];
         a[right]=temp;
         right--;
         left++;
      }
   }
   while(left<=right);
   if(low<right)
   quickSort(a,low,right);
   if(left<high)
   quickSort(a,left,high);
}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Enter array size : 5
Enter 5 elements : 34 67 12 45 22
Before sorting the elements are : 34 67 12 45 22
After sorting the elements are : 12 22 34 45 67

Test Case - 2
User Output
Enter array size : 8
Enter 8 elements : 77 55 22 44 99 33 11 66
Before sorting the elements are : 77 55 22 44 99 33 11 66

ID: 224G1A0516

Srinivasa Ramanujan Institute of Technology

After sorting the elements are : 11 22 33 44 55 66 77 99

Test Case - 3
User Output
Enter array size : 5
Enter 5 elements : -32 -45 -67 -46 -14
Before sorting the elements are : -32 -45 -67 -46 -14
After sorting the elements are : -67 -46 -45 -32 -14