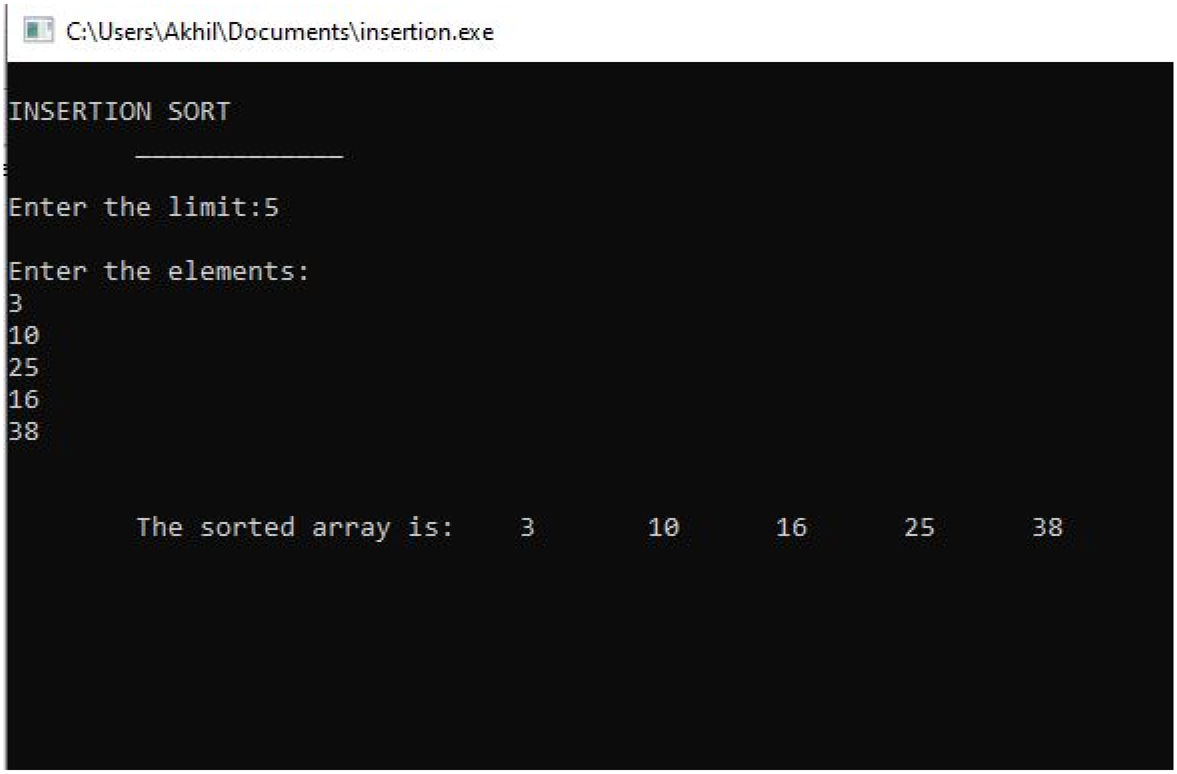
(Q) C program to perform insertion sort ?

Ans :

```
#include<stdio.h>
#include<conio.h>
void main()
{
int a[50],i,n,j,temp;
printf("\nINSERTION SORT\n");
printf("\t______________\n");
printf("\nEnter the limit:");
scanf("%d",&n);
printf("\nEnter the elements:\n");
for(i=0;i<n;i++)
{
scanf("%d",&a[i]);
}
for(i=1;i<n;i++)
{
temp=a[i];
j=i-1;
while(temp<a[j]&&j>=0)
{
a[j+1]=a[j];
j--;
}
a[j+1]=temp;
}
printf("\n\n\tThe sorted array is:");
for(i=0;i<n;i++)
{
printf("\t%d",a[i]);
}
getch();
}
```

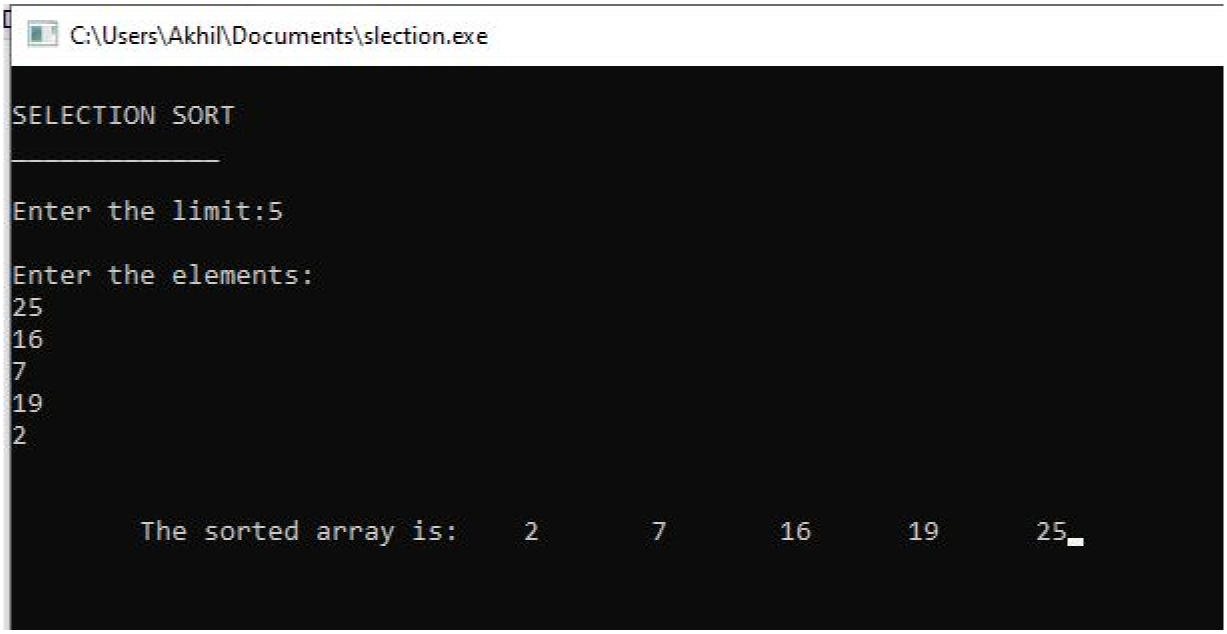


```
INSERTION SORT
        ______________

Enter the limit:5

Enter the elements:
3
10
25
16
38


        The sorted array is:    3       10      16      25      38
```

(Q) C program to perform selection sort ?

Ans :

```
#include<stdio.h>
#include<conio.h>
void main()
{
int a[50],i,n,j,temp;
printf("\nSELECTION SORT");
printf("\n______________\n");
printf("\n Enter the limit:");
scanf("%d",&n);
printf("\n Enter the elements: \n");
for(i=0;i<n;i++)
{
scanf("%d",&a[i]);
}
for(i=0;i<n-1;i++)
{
for(j=i+1;j<n;j++)
{
if(a[i]>a[j])
{
temp=a[i];
a[i]=a[j];
a[j]=temp;
}
}
}
printf("\n\n\tThe sorted array is:");
for(i=0;i<n;i++)
{
printf("\t%d",a[i]);
}
getch();
}
```



```
SELECTION SORT
______________

Enter the limit:5

Enter the elements:
25
16
7
19
2


        The sorted array is:    2       7       16      19      25_
```

(Q) Create a text file containing the name, height, weight of the students in a class. Perform Quick sort ?

Ans :

```
#include<stdio.h>
struct student
{
        char name[100];
        float height;
        int weight;
};
```

```
 struct student s[10],t;
void quicksort(struct student s[10],int,int);
int partition(struct student s[10],int,int);
void mergesort(struct student s[10],int,int);
void merge(struct student s[10],int,int,int);
void main()
{
   struct student s[10];
   FILE *fptr;
   int i;
   fptr=fopen("quicksort.txt","w");
   for(i=0;i<3;++i)
   {
          fflush(stdin);
          printf("enter the name of the students");
          gets(s[i].name);
          printf("\nenter the height\n");
          scanf("%f",&s[i].height);
          printf("\nenter the weight of the students\n");
          scanf("%d",&s[i].weight);
          fprintf(fptr,"\nname:%s\nheight:%.1f\nweight:%d\n",s[i].name,s[i].height,s[i].weight);

          }
          fprintf(fptr,"\n\nquick sort on the basis of weight\n\n");
                    quicksort(s,0,3-1);
           for(i=0;i<3;i++)
           {

         fprintf(fptr,"\nname:%s\nheight:%.1f\nweight:%d\n",s[i].name,s[i].height,s[i].weight);
      }
      fclose(fptr);
      FILE *f1;
      f1=fopen("mergesort.txt","w");
        mergesort(s,0,3-1);
 fprintf(f1,"\n\nData After Merge Sort on the basis of height\n\n");
           for(i=0;i<3;i++)
           {

            fprintf(f1,"\nname:%s\nheight:%.1f\nweight:%d\n",s[i].name,s[i].height,s[i].weight);
}
fclose(f1);
}
void quicksort(struct student s[10],int lb,int ub)
{
 int mid;
 if(lb<ub)
 {
 mid=partition(s,lb,ub);
 quicksort(s,lb,mid-1);
 quicksort(s,mid+1,ub);
 }
}
 int partition(struct student s[10],int lb,int ub)
{
            int p,q,pivot;
 p=lb+1;
 q=ub;
 pivot=s[lb].weight;
 while(q>=p)
 {
 while(s[p].weight<pivot)
 p++;
 while(s[q].weight>pivot)
 q--;
 if(q>p)
 {
 t=s[p];
 s[p]=s[q];
 s[q]=t;
 }
 }
 t=s[lb];
 s[lb]=s[q];
 s[q]=t;
 return q;
}
void mergesort(struct student s[],int lb,int ub)
{
 int mid;
 if(lb<ub)
 {
 mid=(lb+ub)/2;
 mergesort(s,lb,mid);
 mergesort(s,mid+1,ub);
 merge(s,lb,mid+1,ub);
 }
}
void merge(struct student s[10],int lb,int mid,int ub)
```

```
{
 int k,p1,p2,p3;
 struct student s1[10];
 p1=lb;
 p3=lb;
 p2=mid;
 while((p1<mid)&&(p2<=ub))
 {
 if(s[p1].height<=s[p2].height)
 s1[p3++]=s[p1++];
 else
 s1[p3++]=s[p2++];
 }
 while(p1<mid)
 {
 s1[p3++]=s[p1++];
 }
 while(p2<=ub)
 {
 s1[p3++]=s[p2++];
 }
 for(k=lb;k<p3;k++)
 {
 s[k]=s1[k];
 }
}
```