Priority Queue

1. Add the elements into the queue according to the order (ascending or descending).

2. Delete the elements.

```
#include <stdio.h>
#include<stdlib.h>
#define max 50
void inser(int);
void delete(int);
void display();
int pq[max];
int front=-1,rear=-1;
int main()
{
   int ele,ch;
   while(1)
   {
      printf("1.insert\n2.delete\n3.display\n4.exit\n");
      printf("enter choice");
      scanf("%d",&ch);
      switch(ch)
      {
         case 1: printf("enter element");
            scanf("%d",&ele);
            inser(ele);
            break;
         case 2: printf("delete");
```

```
                scanf("%d",&ele);
                delete(ele);
                break;
            case 3: display();
                 break;
            case 4: exit(1);

            default: printf("INvalid choice\n");
        }
    }
    return 0;
}
void inser(int ele)
{
    int i,j,k=0;
    if(rear==max-1)
    printf("Queue is Full\n");
    else
    {
        if(front==-1 && rear==-1)
        {
            rear++;
            front++;
            pq[rear]=ele;
        }
        else
```

```
        {
            for(i=0;i<=rear;i++)
            {
                if(ele>pq[i])
                {   k=1;
                    for(j=rear+1;j>i;j--)
                    {
                        pq[j]=pq[j-1];
                    }
                    pq[i]=ele;
                    rear++;
                    break;
                }
            }
            if(k==0)
            {
                pq[rear+1]=ele;
                rear++;
            }
        }
    }
}
void delete(int data)
 {
    int i;
```

```
    if ((front==-1) && (rear==-1))
    {
        printf("\nQueue is empty no elements to delete");

    }
  else{
    for (i = 0; i <= rear; i++)
    {
        if (data == pq[i])
        {
            for (; i < rear; i++)
            {
                pq[i] = pq[i + 1];
            }

        pq[i] = -99;
        rear--;

        if (rear == -1)
            front = -1;
        return;
        }
    }
  }
    printf("\n%d not found in queue to delete", data);
}
```

```
/* Function to display queue elements */
void display()
{
    if ((front == -1) && (rear == -1))
    {
        printf("\nQueue is empty");
        return;
    }

    for (; front <= rear; front++)
    {
        printf(" %d ", pq[front]);
    }

    front = 0;
}
```