Priority Queue to Add and Delete Elements

```
#include <stdio.h>

#include <stdlib.h>
#define MAX 5

void insert_by_priority(int);

void delete_by_priority(int);

void create();

void check(int);

void display_pqueue();


int pri_que[MAX];

int front, rear;


void main()

{
   int n, ch;

   printf("\n1 - Insert an element into queue");

   printf("\n2 - Delete an element from queue");

   printf("\n3 - Display queue elements");

   printf("\n4 - Exit");

   create();
```

```
    while (1)

    {
        printf("\nEnter your choice : ");
        scanf("%d", &ch);

        switch (ch)

        {
        case 1:
            printf("\nEnter value to be inserted : ");
           scanf("%d",&n);
            insert_by_priority(n);
            break;

        case 2:
            printf("\nEnter value to delete : ");
           scanf("%d",&n);
            delete_by_priority(n);
            break;

        case 3:

            display_pqueue();
            break;

        case 4:
            exit(0);
        default:

            printf("\nChoice is incorrect, Enter a correct choice");

        }

    }

}
```

```
/* Function to create an empty priority queue */

void create()
{
   front = rear = -1;
}

/* Function to insert value into priority queue */

void insert_by_priority(int data)
{
   if (rear >= MAX - 1)
   {
      printf("\nQueue overflow no more elements can be inserted");
      return;
   }
   if ((front == -1) && (rear == -1))
   {
      front++;
      rear++;
      pri_que[rear] = data;
      return;
   }

   else
      check(data);
   rear++;
}


/* Function to check priority and place element */

void check(int data)
{
   int i,j;
   for (i = 0; i <= rear; i++)
   {
      if (data >= pri_que[i])
      {
         for (j = rear + 1; j > i; j--)
```

```
            {
               pri_que[j] = pri_que[j - 1];

            }
            pri_que[i] = data;

            return;

         }

      }

      pri_que[i] = data;

}


/* Function to delete an element from queue */

void delete_by_priority(int data)
{
   int i;
   if ((front==-1) && (rear==-1))
   {
      printf("\nQueue is empty no elements to delete");
      return;
   }

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

      pri_que[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_pqueue()
{
    if ((front == -1) && (rear == -1))

    {
        printf("\nQueue is empty");
        return;

    }

    for (; front <= rear; front++)
    {
        printf(" %d ", pri_que[front]);
    }
    front = 0;

}
```