MULTIPLE PRIORITY QUEUE

CODE:

```
#include<stdio.h>
#define N 3
int queue[3][N];
int front[3]={0,0,0};
int rear[3]={-1,-1,-1};
int item,pr;
void main()
{
int ch;
while(1)
{
printf("PRIORITY QUEUE\n");
printf("*******\n");
printf("\n\t1:PQinsert\n");
printf("\n\t2:PQdelete\n");
printf("\n\t3:PQdisplay\n");
printf("\n\t4:Exit\n");
printf("\nenter the choice\n");
scanf("%d",&ch);
switch(ch)
{
 case 1:printf("\nenter the priority number\n");
                scanf("%d",&pr);
                if(pr>0 && pr<4)
                pqinsert(pr-1);
                else
                printf("\only 3 priority exists 1 2 3\n");
                break;
case 2:pqdelete();
            break;
case 3:display();
            break;
case 4:exit(0);
}
}
}
pqinsert(int pr)
{
 if(rear[pr]==N-1)
 printf("\n Queue overflow\n");
 else
 {
```

```
 printf("\nenter the item\n");
 scanf("%d",&item);
 rear[pr]++;
 queue[pr][rear[pr]]=item;
 }
 return;
}
pqdelete()
{
int i;
for(i=0;i<3;i++)
{
  if(rear[i]==front[i]-1)
  printf("\queue empty\n");
  else
  {
  printf("deleted item is %d of queue %d\n",queue[i][front[i]],i+1);
  front[i]++;
  return;
  }
 }
}
display()
{
int i,j;
for(i=0;i<3;i++)
{
if(rear[i]==front[i]-1)
  printf("\queue empty %d\n",i+1);
else
  {
  printf("\nQUEUE %d:",i+1);
  for(j=front[i];j<=rear[i];j++)
    printf("%d\t",queue[i][j]);
  }
}
  return;
}
```

OUTPUT:

```
PRIORITY QUEUE
*******

        1:PQinsert

        2:PQdelete

        3:PQdisplay

        4:Exit

enter the choice
1

enter the priority number
3

enter the item
1
PRIORITY QUEUE
*******

        1:PQinsert

        2:PQdelete

        3:PQdisplay

        4:Exit

enter the choice
1

enter the priority number
2

enter the item
2
```

```
PRIORITY QUEUE
*******

        1:PQinsert

        2:PQdelete

        3:PQdisplay

        4:Exit

enter the choice
1

enter the priority number
1

enter the item
3
PRIORITY QUEUE
*******

        1:PQinsert

        2:PQdelete

        3:PQdisplay

        4:Exit

enter the choice
3

QUEUE 1:3
QUEUE 2:2
QUEUE 3:1       PRIORITY QUEUE
*******
```

```
*******

        1:PQinsert

        2:PQdelete

        3:PQdisplay

        4:Exit

enter the choice
1

enter the priority number
4
only 3 priority exists 1 2 3
PRIORITY QUEUE
*******

        1:PQinsert

        2:PQdelete

        3:PQdisplay

        4:Exit

enter the choice
3

QUEUE 1:3
QUEUE 2:2
QUEUE 3:1       PRIORITY QUEUE
```

```
        1:PQinsert

        2:PQdelete

        3:PQdisplay

        4:Exit

enter the choice
2
deleted item is 3 of queue 1
PRIORITY QUEUE
*******

        1:PQinsert

        2:PQdelete

        3:PQdisplay

        4:Exit

enter the choice
2
queue empty
deleted item is 2 of queue 2
PRIORITY QUEUE
*******

        1:PQinsert

        2:PQdelete

        3:PQdisplay

        4:Exit

enter the choice
2
queue empty
queue empty
deleted item is 1 of queue 3
```

```
PRIORITY QUEUE
*******

        1:PQinsert

        2:PQdelete

        3:PQdisplay

        4:Exit

enter the choice
2
queue empty
queue empty
queue empty
PRIORITY QUEUE
*******

        1:PQinsert

        2:PQdelete

        3:PQdisplay

        4:Exit

enter the choice
4

Process returned 0 (0x0)   execution time : 54.834 s
Press any key to continue.
```

ASCENDING PRIORITY QUEUE:
CODE:

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#define MAX 3

int pq[MAX];
int count = 0;
int d = 0;

void insert(int data){
  int i = 0;
    if(count==MAX)
    {
```

```
        printf("Queue overflow\n");
        return;
    }
    if(count == 0){
      pq[count++] = data;
    }else{
                          for(i = count - 1; i >= 0; i-- ){
        if(data<pq[i]){
          pq[i+1] = pq[i];
        }else{
          break;
        }
      }


      pq[i+1] = data;
      count++;
    }

}

int removeData(){

  return pq[d++];
}
void display()
{int i;
if (count==0)
{
        printf("queue is empty\n");
        return;
}
printf("Contents of queue: ");
for(i=d;i<count;i++)
{
        printf("%d ",pq[i]);
}
printf("\n");
}

int main() {
   int choice,item;
        for(;;)
        {
```

```
            printf("\n1:insert\n2:delete_smallest\n3:display\n4:exit\n");
            printf("Enter the choice :");
            scanf("%d",&choice);
            switch(choice)
            {
                  case 1:printf("Enter the item to be inserted :");
                  scanf("%d",&item);
                  insert(item);
                  break;
                  case 2:item=removeData();
                  if(item==-1)
                  printf("Queue is empty\n");
                  else
                  printf("item deleted=%d\n",item);
                  break;
                  case 3:display();
                  break;
                  default:exit (0);

            }

      }

}
```

OUTPUT:

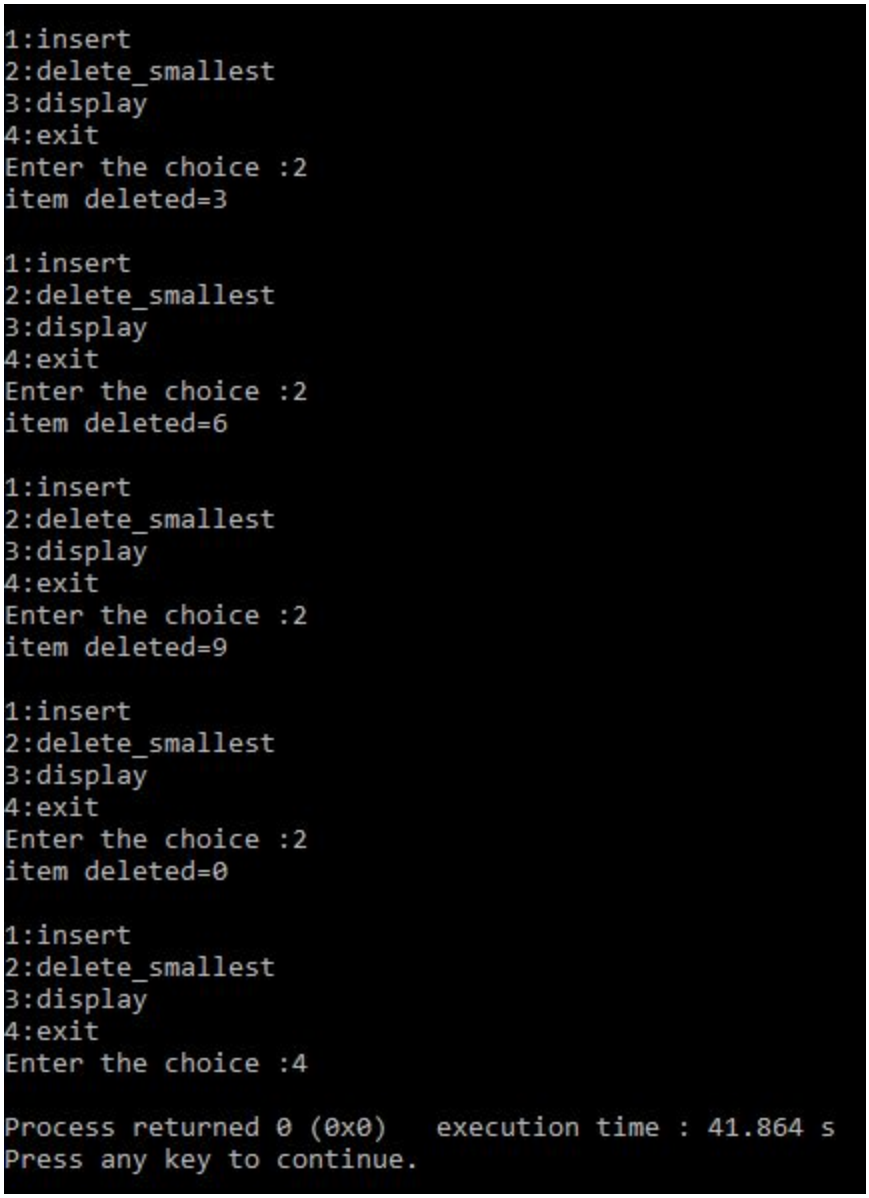
```
1:insert
2:delete_smallest
3:display
4:exit
Enter the choice :1
Enter the item to be inserted :6

1:insert
2:delete_smallest
3:display
4:exit
Enter the choice :1
Enter the item to be inserted :3

1:insert
2:delete_smallest
3:display
4:exit
Enter the choice :1
Enter the item to be inserted :9

1:insert
2:delete_smallest
3:display
4:exit
Enter the choice :1
Enter the item to be inserted :7
Queue overflow

1:insert
2:delete_smallest
3:display
4:exit
Enter the choice :3
Contents of queue: 3 6 9
```



DESCENDING PRIORITY QUEUE:

CODE:

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#define MAX 3

int pq[MAX];
int count = 0;
int d = 0;

void insert(int data){
```

```
  int i = 0;
    if(count==MAX)
    {
       printf("Queue overflow\n");
       return;
    }
    if(count == 0){
      pq[count++] = data;
    }else{
                          for(i = count - 1; i >= 0; i-- ){
        if(data>pq[i]){
          pq[i+1] = pq[i];
        }else{
          break;
        }
      }


      pq[i+1] = data;
      count++;
    }

}

int removeData(){

  return pq[d++];
}
void display()
{int i;
if (count==0)
{
        printf("queue is empty\n");
        return;
}
printf("Contents of queue: ");
for(i=d;i<count;i++)
{
        printf("%d ",pq[i]);
}
printf("\n");
}

int main() {
```

```
int choice,item;
        for(;;)
        {
                printf("\n1:insert\n2:delete_largest\n3:display\n4:exit\n");
                printf("Enter the choice :");
                scanf("%d",&choice);
                switch(choice)
                {
                        case 1:printf("Enter the item to be inserted :");
                        scanf("%d",&item);
                        insert(item);
                        break;
                        case 2:item=removeData();
                        if(item==-1)
                        printf("Queue is empty\n");
                        else
                        printf("item deleted=%d\n",item);
                        break;
                        case 3:display();
                        break;
                        default:exit (0);

                }

        }

}
```

OUTPUT:

```
1:insert
2:delete_largest
3:display
4:exit
Enter the choice :1
Enter the item to be inserted :7

1:insert
2:delete_largest
3:display
4:exit
Enter the choice :1
Enter the item to be inserted :5

1:insert
2:delete_largest
3:display
4:exit
Enter the choice :1
Enter the item to be inserted :9

1:insert
2:delete_largest
3:display
4:exit
Enter the choice :1
Enter the item to be inserted :5
Queue overflow

1:insert
2:delete_largest
3:display
4:exit
Enter the choice :3
Contents of queue: 9 7 5
```

```
1:insert
2:delete_largest
3:display
4:exit
Enter the choice :2
item deleted=9

1:insert
2:delete_largest
3:display
4:exit
Enter the choice :2
item deleted=7

1:insert
2:delete_largest
3:display
4:exit
Enter the choice :2
item deleted=5

1:insert
2:delete_largest
3:display
4:exit
Enter the choice :2
item deleted=0

1:insert
2:delete_largest
3:display
4:exit
Enter the choice :4

Process returned 0 (0x0)   execution time : 47.939 s
Press any key to continue.
```