Multiple priority queue:

```c
#include<stdio.h>
#include<stdlib.h>
#include<conio.h>
#define N 3
void pqinsert(int);
void pqdelete();
void display();
int queue[3][N];
int front[3]={0,0,0};
int rear[3]={-1,-1,-1};
int item,pr;
int main()
{
int ch;
while(1)
{
printf("\nPRIORITY 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("\n enter the priority number\n");
        scanf("%d",&pr);
        if(pr>0 && pr<4)
        pqinsert(pr-1);
        pqinsert(pr-1);
        else
        printf("\n only 3 priority exists 1 2 3\n");
        break;
case 2:pqdelete();
       break;
case 3:display();
       break;
case 4:exit(0);
}
}
getch();
}
 void pqinsert(int pr)
{
 if(rear[pr]==N-1)
 printf("\n Queue overflow\n");
 else
 {
 printf("\n enter the item\n");
 scanf("%d",&item);
 rear[pr]++;
 queue[pr][rear[pr]]=item;
 }

}
void pqdelete()
{
int i;
for(i=0;i<3;i++)
```

```
int i;
for(i=0;i<3;i++)
{
  if(rear[i]==front[i]-1)
  printf("\n queue empty\n");
  else
  {
  printf("deleted item is %d of queue %d\n",queue[i][front[i]],i+1);
  front[i]++;

  }
 }
}
void display()
{
int i,j;
for(i=0;i<3;i++)
{
if(rear[i]==front[i]-1)
  printf("\n 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]);
  }
}

}
```

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

        1:PQinsert

        2:PQdelete

        3:PQdisplay

        4:Exit

enter the choice
1

 enter the priority number
2

 enter the item
6

PRIORITY QUEUE
*******

        1:PQinsert

        2:PQdelete

        3:PQdisplay
```

```
        3:PQdisplay

        4:Exit

enter the choice
1

 enter the priority number
3

 enter the item
8

PRIORITY QUEUE
*******

        1:PQinsert

        2:PQdelete

        3:PQdisplay

        4:Exit

enter the choice
1

 enter the priority number
1
```

```
 enter the priority number
1

 enter the item
4

PRIORITY QUEUE
*******

        1:PQinsert

        2:PQdelete

        3:PQdisplay

        4:Exit

enter the choice
3

QUEUE 1:4
QUEUE 2:6
QUEUE 3:8
PRIORITY QUEUE
*******

        1:PQinsert

        2:PQdelete
```

```
        2:PQdelete

        3:PQdisplay

        4:Exit

enter the choice
2
deleted item is 4 of queue 1
deleted item is 6 of queue 2
deleted item is 8 of queue 3

PRIORITY QUEUE
*******

        1:PQinsert

        2:PQdelete

        3:PQdisplay

        4:Exit

enter the choice
3

 queue empty 1

 queue empty 2
```

```
 queue empty 2

 queue empty 3

PRIORITY QUEUE
*******

        1:PQinsert

        2:PQdelete

        3:PQdisplay

        4:Exit

enter the choice
4


...Program finished with exit code 0
Press ENTER to exit console.
```

PRIORITY QUEUE:

```c
#include<stdio.h>
#include<stdlib.h>
struct node
{
        int priority;
        int info;
        struct node *link;
}*front=NULL;
void insert(int item, int item_priority);
int del();
void display();
int isEmpty();
int main()
{
        int choice,item,item_priority;
        while(1)
        {
                printf("\n1.Insert\n");
                printf("2.Delete\n");
                printf("3.Display\n");
                printf("4.Quit\n");
                printf("\nEnter your choice : ");
                scanf("%d", &choice);
                switch(choice)
                {
                 case 1:
                        printf("\nInput the item to be added in the queue : ");
                        scanf("%d",&item);
                        printf("\nEnter its priority : ");
                        scanf("%d",&item_priority);
                        insert(item, item_priority);
                        break;
                 case 2:
                        printf("\nDeleted item is %d\n",del());
                        break;
                 case 3:
                        display();
                        break;
                 case 4:
                        exit(1);
                 default :
                        printf("\nWrong choice\n");
                }
        }

        return 0;
}
void insert(int item,int item_priority)
{
        struct node *tmp,*p;

        tmp=(struct node *)malloc(sizeof(struct node));
        if(tmp==NULL)
        {
                printf("\nMemory not available\n");
                return;
        }
        tmp->info=item;
        tmp->priority=item_priority;

        if( isEmpty() || item_priority < front->priority )
```

```
        if( isEmpty() || item_priority < front->priority )
        {
                tmp->link=front;
                front=tmp;
        }
        else
        {
                p = front;
                while( p->link!=NULL && p->link->priority<=item_priority )
                        p=p->link;
                tmp->link=p->link;
                p->link=tmp;
        }
}
int del()
{
        struct node *tmp;
        int item;
        if( isEmpty() )
        {
                printf("\nQueue Underflow\n");
                exit(1);
        }
        else
        {
                tmp=front;
                item=tmp->info;
                front=front->link;
                free(tmp);
        }
        return item;
}
```

```
int isEmpty()
{
        if( front == NULL )
                return 1;
        else
                return 0;
}
void display()
{
        struct node *ptr;
        ptr=front;
        if( isEmpty() )
                printf("\nQueue is empty\n");
        else
        {       printf("\nQueue is :\n");
                printf("\nPriority        Item\n");
                while(ptr!=NULL)
                {
                        printf("%5d         %5d\n",ptr->priority,ptr->info);
                        ptr=ptr->link;
                }
        }
}
```

```
1.Insert
2.Delete
3.Display
4.Quit

Enter your choice : 1

Input the item to be added in the queue : 2

Enter its priority : 3

1.Insert
2.Delete
3.Display
4.Quit

Enter your choice : 1

Input the item to be added in the queue : 5

Enter its priority : 2

1.Insert
2.Delete
3.Display
4.Quit
```

```
4.Quit

Enter your choice : 1

Input the item to be added in the queue : 4

Enter its priority : 2

1.Insert
2.Delete
3.Display
4.Quit

Enter your choice : 1

Input the item to be added in the queue : 3

Enter its priority : 3

1.Insert
2.Delete
3.Display
4.Quit

Enter your choice : 3

Queue is :
```

```
Queue is :

Priority       Item
    2            5
    2            4
    3            2
    3            3

1.Insert
2.Delete
3.Display
4.Quit

Enter your choice : 2

Deleted item is 5

1.Insert
2.Delete
3.Display
4.Quit

Enter your choice : 1

Input the item to be added in the queue : 6

Enter its priority : 1
```

```
Enter its priority : 1

1.Insert
2.Delete
3.Display
4.Quit

Enter your choice : 3

Queue is :

Priority       Item
    1            6
    2            4
    3            2
    3            3

1.Insert
2.Delete
3.Display
4.Quit

Enter your choice : 2

Deleted item is 6

1.Insert
2.Delete
```

```
3.Display
4.Quit

Enter your choice : 2

Deleted item is 4

1.Insert
2.Delete
3.Display
4.Quit

Enter your choice : 3
```

```
Queue is :

Priority        Item
    3             2
    3             3

1.Insert
2.Delete
3.Display
4.Quit

Enter your choice : 4


...Program finished with exit code 1
Press ENTER to exit console.
```

Dequeue:

```c
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
#define qsize 5
int f=0,r=-1,ch;
int item,q[10];

int isfull()
  {
   return(r==qsize-1)?1:0;
  }
int isempty()
  {
   return(f>r)?1:0;
  }
void insert_rear()
  {
   if(isfull())
{
 printf("queue overflow\n");
 return;
}
   r=r+1;
   q[r]=item;
  }
void delete_front()
  {
   if(isempty())
{
 printf("queue empty\n");
```

```c
 printf("queue empty\n");
 return;
}
   printf("item deleted is %d\n",q[(f)++]);
   if(f>r)
{
 f=0;
 r=-1;
}
  }
void insert_front()
  {
   if(f!=0)
{
 f=f-1;
 q[f]=item;
 return;
}
else if((f==0)&&(r==-1))
{
 q[++(r)]=item;
 return;
}
else
 printf("insertion not possible\n");
  }
void delete_rear()
  {
   if(isempty())
{
 printf("queue is empty\n");
```

```
 printf("queue is empty\n");
 return;
}
   printf("item deleted is %d\n",q[(r)--]);
   if(f>r)
{
 f=0;
 r=-1;
}
  }
void display()
  {
   int i;
   if(isempty())
{
 printf("queue empty\n");
 return;
}
   for(i=f;i<=r;i++)
printf("%d\n",q[i]);
  }
void main()
  {for(;;)
   {
printf("1.insert_rear\n2.insert_front\n3.delete_rear\n4.delete_front\n5.display\n6.exit\n");
printf("enter choice\n");
scanf("%d",&ch);
switch(ch)
 {
  case 1:printf("enter the item\n");
 scanf("%d",&item);
```

```
  case 1:printf("enter the item\n");
 scanf("%d",&item);
 insert_rear();
 break;
  case 2:printf("enter the item\n");
 scanf("%d",&item);
 insert_front();
 break;
  case 3:delete_rear();
 break;
  case 4:delete_front();
 break;
  case 5:display();
 break;
  default:exit(0);
 }}}
```

```
enter the item
6
insertion not possible
1.insert_rear
2.insert_front
3.delete_rear
4.delete_front
5.display
6.exit
enter choice
1
enter the item
9
1.insert_rear
2.insert_front
3.delete_rear
4.delete_front
5.display
6.exit
enter choice
5
2
9
1.insert_rear
2.insert_front
3.delete_rear
4.delete_front
5.display
6.exit
enter choice
4
```

```
enter choice
4
item deleted is 2
1.insert_rear
2.insert_front
3.delete_rear
4.delete_front
5.display
6.exit
enter choice
4
item deleted is 9
1.insert_rear
2.insert_front
3.delete_rear
4.delete_front
5.display
6.exit
```

```
enter choice
5
queue empty
1.insert_rear
2.insert_front
3.delete_rear
4.delete_front
5.display
6.exit
enter choice
6


...Program finished with exit code 0
Press ENTER to exit console.
```