

Multiple Priority Queue

```
#include <stdio.h>
```

```
#include <stdlib.h>
```

```
#define N 3
```

```
int queue [3][N];
```

```
int front [3] = {0, 0, 0};
```

```
int rear [3] = {-1, -1, -1};
```

```
int item, pr;
```

```
int pp main()
```

```
{
```

```
int ch;
```

```
while (1)
```

```
{
```

```
printf("\n\t 1: PQinsert\n");
```

```
printf("*****\n");
```

```
printf("\t 1: PQinsert\n");
```

```
printf("\t 2: PQdelete\n");
```

```
printf("\t 3: Display\n");
```

```
printf("\t 4: Exit\n");
```

```
printf("Enter the choice\n");
```

```
scanf("%d", &ch);
```

```
switch (ch)
```

```
{
```

```
Case 1: printf("Enter the priority number\n");
```

```
scanf("%d", &pr);
```

```
if (pr > 0 && pr < 4)
```

```
PQinsert(pr-1);
```

```
else
```

```
printf("only 3 priority exists\n");
```

```
break;
```

```
Case 2: PQdelete();
```

```
break;
```

```
case 3: display();
```

```
break;
```

```
case 4: exit(0);
```

```
}
```

```
}
```

```
}
```

```
void insert (int pr) {
```

```
if (rear[pr] == N-1)
```

```
printf ("Queue Overflow \n");
```

```
else
```

```
{
```

```
printf ("Enter the item \n");
```

```
scanf ("%d", &item);
```

```
rear[pr]++;
```

```
queue[pr][rear[pr]] = item;
```

```
}
```

```
}
```

```
void delete ()
```

```
{
```

```
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 %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("Queue empty %d \n", i+1);

else

{

printf("In queue %d :", i+1);

for (j = front[i]; j <= rear[i]; j++)

printf("%d \t", q[i][j]);

}

}

}

Ascending Priority Queue

```
#include <stdio.h>
#define que_size 3
int item, front = 0, rear = -1, q[que_size];
void insertrear()
{
    if (rear == que_size - 1)
    {
        printf("Queue Overflow\n");
        return;
    }
    int i, j, min;
    printf("Enter the item\n");
    scanf("%d", &item);
    q[rear + 1] = item;
    if (rear >= 1)
    {
        for (i = 1; i <= rear; i++)
        {
            min = q[i];
            j = i - 1;
            while (j >= 0 && q[j] > min)
            {
                q[j + 1] = q[j];
                j--;
            }
            q[j + 1] = min;
        }
    }
}

int deletefront()
{
    if (front <= rear)
```



```
if (front > rear)
```

```
{ front = 0;
```

```
rear = -1;
```

```
return -1;
```

```
} return q[front++];
```

```
} void display()
```

```
{ int i;
```

```
if (front > rear)
```

```
{ printf("Queue is empty\n");  
return;
```

```
} printf("Contents of queue\n");
```

```
for (i = front; i <= rear; i++)
```

```
{ printf("%d\n", q[i]);
```

```
}
```

```
} int main()
```

```
{ int choice;
```

```
for (;;) 
```

```
{ printf("\n 1. Insert rear 2. Delete front 3. display 4. Exit\n");
```

```
printf("Enter choice: ");
```

```
scanf("%d", &choice);
```

switch (choice)

1

case 1: insertrear();
break;

case 2: item = deletefront();

if (item == -1)

printf("Queue is empty\n");

else

printf("item deleted is %d\n", item);

break;

case 3: display();

break;

default: exit(0);

}
}
}

```

42     else
43     {
44         printf("\nQUEUE %d:",i+1);
45         for(j=front[i];j<=rear[i];j++)
46             printf("%d\t",queue[i][j]);
47     }
48 }
49
50 }
51 int main()
52 {
53     int ch;
54     while(1)
55     {
56         printf("\n\t1:PQinsert\n");
57         printf("\n\t2:PQdelete\n");
58         printf("\n\t3:PQdisplay\n");
59         printf("\n\t4:Exit\n");
60         printf("\nenter the choice\n");
61         scanf("%d",&ch);
62         switch(ch)
63         {
64             case 1:printf("\nenter the priority
65                        number\n");
66                     scanf("%d",&pr);
67                     if(pr>0 && pr<4)
68                         pqinsert(pr-1);
69                     else
70                         printf("\nonly 3 priority exists 1 2
71                        3\n");
72                     break;
73             case 2: pqdelete();
74                     break;
75             case 3: display();
76                     break;
77             case 4:exit(0);
78         }
79     }
80 }

```



```

1  #include<stdio.h>
2  #include<stdlib.h>
3  #define N 5
4  int queue[3][N];
5  int front[3]={0,0,0};
6  int rear[3]={-1,-1,-1};
7  int item,pr;
8
9  int pqinsert(int pr)
10 {
11     if(rear[pr]==N-1)
12         printf("\n Queue overflow\n");
13     else
14     {
15         printf("\n enter the item\n");
16         scanf("%d",&item);
17         rear[pr]++;
18         queue[pr][rear[pr]]=item;
19     }
20 }
21 int pqdelete()
22 {
23     int i;
24     for(i=0;i<3;i++)
25     {
26         if(rear[i]==front[i]-1)
27             printf("\nqueue empty\n");
28         else
29         {
30             printf("deleted item is %d of queue %d\n",
31 queue[i][front[i]],i+1);
32             front[i]++;
33         }
34     }
35 }
36 int display()
37 {
38     int i,j;
39     for(i=0;i<3;i++)
40     {
41         if(rear[i]==front[i]-1)
42             printf("\nqueue %d is empty\n",i+1);

```


enter the choice

1

enter the priority number

1

enter the item

20

1:PQinsert

2:PQdelete

3:PQdisplay

4:Exit

enter the choice

1

enter the priority number

2

enter the item

100

1:PQinsert

2:PQdelete

3:PQdisplay

4:Exit

enter the choice

1

enter the priority number

3

enter the item

1000

1:PQinsert

2:PQdelete

3:PQdisplay

4:Exit

enter the choice

3

QUEUE 1:10 20

QUEUE 2:100

QUEUE 3:1000

1:PQinsert

2:PQdelete

3:PQdisplay

4:Exit

enter the choice

2

deleted item is 10 of queue 1

deleted item is 100 of queue 2

deleted item is 1000 of queue 3

1:PQinsert

2:PQdelete

3:PQdisplay

4:Exit

enter the choice

3

QUEUE 1:20

queue 2 is empty

queue 3 is empty

1:PQinsert

2:PQdelete

3:PQdisplay

4:Exit

enter the choice

```

1  #include <stdio.h>
2  #include <string.h>
3  #include <stdlib.h>
4  #define MAX 4
5
6  int pq[MAX];
7  int count = 0;
8  int d = 0;
9
10 void insert(int data){
11     int i = 0;
12     if(count==MAX)
13     {
14         printf("Queue overflow\n");
15         return;
16     }
17     // if queue is empty, insert the data
18     if(count == 0){
19         pq[count++] = data;
20     }else{
21         // start from the right end of the
22         // queue
23         for(i = count - 1; i >= 0; i-- ){
24             //if data is smaller shift right
25             if(data<pq[i]){
26                 pq[i+1] = pq[i];
27             }else{
28                 break;
29             }
30         }
31         // insert the data
32         pq[i+1] = data;
33         count++;
34     }
35 }
36
37
38 int removeData(){
39
40     return pq[d++];
41 }

```

```

42 void display()
43 {int i;
44 if (count==0)
45 {
46     printf("queue is empty\n");
47     return;
48 }
49 printf("Contents of queue: ");
50 for(i=d;i<count;i++)
51 {
52     printf("%d ",pq[i]);
53 }
54 printf("\n");
55 }
56
57 int main() {
58     int choice,item;
59     for(;;)
60     {
61         printf("\n1:insert 2:delete_smallest
62         3:display 4:exit\n");
63         printf("Enter the choice :");
64         scanf("%d",&choice);
65         switch(choice)
66         {
67             case 1:printf("Enter the item to be

```



```
67     scanf("%d",&item);
68     insert(item);
69     break;
70     case 2:item=removeData();
71     if(item== -1)
72     printf("Queue is empty\n");
73     else
74     printf("item deleted=%d\n",item);
75     break;
76     case 3:display();
77     break;
78     default:exit (0);
79
80     }
81 |
82     }
83
84 }
```

1:insert 2:delete_smallest 3:display 4:exit

Enter the choice :1

Enter the item to be inserted :20

1:insert 2:delete_smallest 3:display 4:exit

Enter the choice :1

Enter the item to be inserted :30

1:insert 2:delete_smallest 3:display 4:exit

Enter the choice :1

Enter the item to be inserted :10

1:insert 2:delete_smallest 3:display 4:exit

Enter the choice :3

Contents of queue: 10 20 30

1:insert 2:delete_smallest 3:display 4:exit

Enter the choice :2

item deleted=10

1:insert 2:delete_smallest 3:display 4:exit

Enter the choice :3

Contents of queue: 20 30

1:insert 2:delete_smallest 3:display 4:exit

Enter the choice :■