

```

1 #include <stdio.h>
2 #include<stdlib.h>
3 #define MAX_SIZE 5
4
5 int Pque[MAX_SIZE];
6 int n=-1;
7 void enqueue(int);
8 int dequeue();
9 void display();
10 int main(int argc, char **argv)
11 {
12     int option, item;
13     do{
14
15         printf("\n1. Enqueue\n");
16         printf("2. Dequeue\n");
17         printf("3. Display\n");
18         printf("4. Exit\n");
19         printf("Enter the option:");
20         scanf("%d",&option);
21         switch(option)
22         {
23             case 1: printf("\nEnter the item:");
24                     scanf("%d",&item);
25                     enqueue(item);
26                     break;
27             case 2: item=dequeue();
28                     printf("Removed element is : %d\n",item);
29                     break;
30             case 3: display();
31                     break;
32             case 4: exit(0);
33         }
34     }while (option!=4);
35 }

```

```

36
37 - void enqueue(int item) {
38     // Check if the queue is full
39 -     if (n == MAX_SIZE - 1) {
40         printf("%s\n", "ERROR: Queue is full");
41         return;
42     }
43     n++;
44     Pque[n] = item;
45 }
46
47 // removes the item with the maximum priority
48 // search the maximum item in the array and replace it with
49 // the last item
50 - int dequeue() {
51     int item;
52     // Check if the queue is empty
53 -     if (n == -1) {
54         printf("%s\n", "ERROR: Queue is empty");
55         return -999999;
56     }
57     int i, max = 0;
58     // find the maximum priority
59 -     for (i = 1; i <= n; i++) {
60 -         if (Pque[max] < Pque[i]) {
61             max = i;
62         }
63     }
64     item = Pque[max];
65
66     // replace the max with the last element
67     Pque[max] = Pque[n];
68     n = n - 1;
69     return item;
70 }
71

```



```
// Removes the item with the maximum priority  
// search the maximum item in the array and replace it with  
// the last item
```

```
int dequeue() {  
    int item;  
    // Check if the queue is empty  
    if (n == -1) {  
        printf("%s\n", "ERROR: Queue is empty");  
        return -999999;  
    }  
    int i, max = 0;  
    // find the maximum priority  
    for (i = 1; i <= n; i++) {  
        if (Pque[max] < Pque[i]) {  
            max = i;  
        }  
    }  
    item = Pque[max];  
  
    // replace the max with the last element  
    Pque[max] = Pque[n];  
    n = n - 1;  
    return item;  
}
```

```
void display()  
{  
    int i;  
    if(n==-1)  
        printf("Queue is empty");  
    printf("The Content:");  
    for(i=0;i<=n;i++)  
        printf(" %d",Pque[i]);  
}
```

1. Enqueue

2. Dequeue

3. Display

4. Exit

Enter the option:1

Enter the item:1

1. Enqueue

2. Dequeue

3. Display

4. Exit

Enter the option:1

Enter the item:2

1. Enqueue

2. Dequeue

3. Display

4. Exit

Enter the option:1

Enter the item:3

1. Enqueue

2. Dequeue

3. Display

4. Exit

Enter the option:1

Enter the item:4

1. Enqueue

2. Dequeue

3. Display

Enter the option:1

Enter the item:5

1. Enqueue
2. Dequeue
3. Display
4. Exit

Enter the option:1

Enter the item:6

ERROR: Queue is full

1. Enqueue
2. Dequeue
3. Display
4. Exit

Enter the option:3

The Content: 1 2 3 4 5

1. Enqueue
2. Dequeue
3. Display
4. Exit

Enter the option:2

Removed element is : 5

1. Enqueue
2. Dequeue
3. Display
4. Exit

Enter the option:3

The Content: 1 2 3 4

1. Enqueue
2. Dequeue
3. Display

The Content: 1 2 3 4 5

1. Enqueue
2. Dequeue
3. Display
4. Exit

Enter the option:2

Removed element is : 5

1. Enqueue
2. Dequeue
3. Display
4. Exit

Enter the option:3

The Content: 1 2 3 4

1. Enqueue
2. Dequeue
3. Display
4. Exit

Enter the option:2

Removed element is : 4

1. Enqueue
2. Dequeue
3. Display
4. Exit

Enter the option:3

The Content: 1 2 3

1. Enqueue
2. Dequeue
3. Display
4. Exit

Enter the option:4

...Program finished with exit code 0