```c
#include <stdio.h>
#include<stdlib.h>
#define MAX_SIZE 5

int Pque[MAX_SIZE];
int n=-1;
void enqueue(int);
int dequeue();
void display();
int main(int argc, char **argv)
{
    int option, item;
    do{

        printf("\n1. Enqueue\n");
        printf("2. Dequeue\n");
        printf("3. Display\n");
        printf("4. Exit\n");
        printf("Enter the option:");
        scanf("%d",&option);
        switch(option)
        {
            case 1: printf("\nEnter the item:");
                    scanf("%d",&item);
                    enqueue(item);
                    break;
             case 2: item=dequeue();
                     printf("Removed element is : %d\n",item);
                     break;
            case 3: display();
                    break;
            case 4: exit(0);
        }
    }while (option!=4);
}
```

```c
void enqueue(int item) {
        // Check if the queue is full
        if (n == MAX_SIZE - 1) {
                printf("%s\n", "ERROR: Queue is full");
                return;
        }
        n++;
        Pque[n] = item;
}

// 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;
}
```

```
// 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
```