Dynamic Queue allocation

```
#include <stdio.h>
#include<stdlib.h>
#define MAX 6
typedef struct
{
   char element;
}QUEUE;
QUEUE *Q,*newQ;
int capacity=2;
void Queuealloc(int *front, int *rear)
{

   newQ=calloc(2*capacity,sizeof(QUEUE));
   printf("memory sucessfully allocated\n");
  // capacity=capacity*2;
   int start=(*front+1)%capacity;
   int i,r=0;
   if(start<2)
   {
      for( i=start;i<capacity;i++,r++)
      newQ[r].element=Q[i].element;
   }
   else
   {
      for( i=start;i<capacity;i++,r++)
      newQ[r].element=Q[i].element;
      r=capacity-*front-1;
      for( i=0;i<=*rear;i++,r++)
      newQ[r].element=Q[i].element;
      //for(int k=*rear;i<*front;i++,k++)
      //newQ[i].element=Q[k].element;

   }
   *front=2*capacity-1;
   *rear=capacity-2;
   capacity*=2;
   free(Q);
   Q=newQ;


}
int IsEmpty(int *front,int *rear)
{
   if (*rear==*front)
   return 1;
   else return 0;
}
```

```
int IsFull(int *front,int *rear)
{

   if(*front==(*rear+1)%capacity)
   return 1;
   else
   return 0;
}

void Addq(char item,int *front,int *rear)
{
   if(IsFull(front,rear))
   {
   printf("sorry queue is full\n");
   Queuealloc(front,rear);
   }
   else{
   *rear=(*rear+1)%capacity;
   Q[*rear].element=item;
   }
}
void DeleteQ(int *front,int *rear)
{
   if(IsEmpty(front,rear))
   {
   printf("sorry queue is empty\n");

   }
   else
   {
      *front=(*front+1)%capacity;
   printf("the element deleted from q is %d\n",Q[*front].element);
   }
}
int main()
{

   Q=calloc(2*capacity,sizeof(QUEUE));
   printf("Hello World");
  int rear=0,front=0;
  int t;
  int opt;
  do
  {
     printf("Press 1. ADDQ   2. DELETE Q    3. DISPLAY Q\n");
     scanf("%d",&opt);
     switch(opt)
     {
          case 1: printf("enter the element to be added to queue\n");
                 char e;
                 scanf(" %c",&e);
                 Addq(e,&front,&rear);
                 break;
           case 2: printf("deleting from queue\n");
                  DeleteQ(&front,&rear);

                  break;

           case 3:printf("the elements of the queue are \n");
                  for(t=(front+1)%capacity;t!=rear;t++)
                  printf("%c ",Q[t].element);
                printf("%c ",Q[t].element);
      }
   }while(opt!=4);
    return 0;
}
```

```
Hello WorldPress 1. ADDQ   2. DELETE Q    3. DISPLAY Q
1
enter the element to be added to queue
A
Press 1. ADDQ   2. DELETE Q    3. DISPLAY Q
1
enter the element to be added to queue
B
sorry queue is full
memory sucessfully allocated
Press 1. ADDQ   2. DELETE Q    3. DISPLAY Q
1
enter the element to be added to queue
B
Press 1. ADDQ   2. DELETE Q    3. DISPLAY Q
1
enter the element to be added to queue
C
Press 1. ADDQ   2. DELETE Q    3. DISPLAY Q
1
enter the element to be added to queue
D
sorry queue is full
memory sucessfully allocated
Press 1. ADDQ   2. DELETE Q    3. DISPLAY Q
1
enter the element to be added to queue
E
Press 1. ADDQ   2. DELETE Q    3. DISPLAY Q
1
```

```
enter the element to be added to queue
E
Press 1. ADDQ   2. DELETE Q    3. DISPLAY Q
3
the elements of the queue are
A B C E E Press 1. ADDQ   2. DELETE Q    3. DISPLAY Q
1
enter the element to be added to queue
F
Press 1. ADDQ   2. DELETE Q    3. DISPLAY Q
1
enter the element to be added to queue
G
Press 1. ADDQ   2. DELETE Q    3. DISPLAY Q
1
enter the element to be added to queue
H
sorry queue is full
memory sucessfully allocated
Press 1. ADDQ   2. DELETE Q    3. DISPLAY Q
1
enter the element to be added to queue
G
Press 1. ADDQ   2. DELETE Q    3. DISPLAY Q
3
the elements of the queue are
A B C E E F G G Press 1. ADDQ   2. DELETE Q    3. DISPLAY Q
1
enter the element to be added to queue
H
Press 1. ADDQ   2. DELETE Q    3. DISPLAY Q
1
enter the element to be added to queue
I
Press 1. ADDQ   2. DELETE Q    3. DISPLAY Q
2
deleting from queue
the element deleted from q is 65
Press 1. ADDQ   2. DELETE Q    3. DISPLAY Q
3
the elements of the queue are
B C E E F G G H I Press 1. ADDQ   2. DELETE Q    3. DISPLAY Q
1
enter the element to be added to queue
J
Press 1. ADDQ   2. DELETE Q    3. DISPLAY Q
1
enter the element to be added to queue
```

```
K
Press 1. ADDQ   2. DELETE Q    3. DISPLAY Q
1
enter the element to be added to queue
L
Press 1. ADDQ   2. DELETE Q    3. DISPLAY Q
1
enter the element to be added to queue
M
Press 1. ADDQ   2. DELETE Q    3. DISPLAY Q
1
enter the element to be added to queue
N
Press 1. ADDQ   2. DELETE Q    3. DISPLAY Q
1
enter the element to be added to queue
O
Press 1. ADDQ   2. DELETE Q    3. DISPLAY Q
1
enter the element to be added to queue
P
sorry queue is full
memory sucessfully allocated
Press 1. ADDQ   2. DELETE Q    3. DISPLAY Q
1
enter the element to be added to queue
Q
Press 1. ADDQ   2. DELETE Q    3. DISPLAY Q
3
the elements of the queue are
B C E E F G G H I J K L M N O Q Press 1. ADDQ   2. DELETE Q    3. DISPLAY Q
2
deleting from queue
the element deleted from q is 66
Press 1. ADDQ   2. DELETE Q    3. DISPLAY Q
3
the elements of the queue are
C E E F G G H I J K L M N O Q Press 1. ADDQ   2. DELETE Q    3. DISPLAY Q
1
enter the element to be added to queue
R
Press 1. ADDQ   2. DELETE Q    3. DISPLAY Q
3
the elements of the queue are
C E E F G G H I J K L M N O Q R Press 1. ADDQ   2. DELETE Q    3. DISPLAY Q
2
deleting from queue
the element deleted from q is 67
```

```
Press 1. ADDQ   2. DELETE Q    3. DISPLAY Q
3
the elements of the queue are
E E F G G H I J K L M N O Q R Press 1. ADDQ   2. DELETE Q    3. DISPLAY Q
```