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

typedef int QueueElementType;

typedef struct QueueNode *QueuePointer;

typedef struct QueueNode
{
    QueueElementType Data;
    QueuePointer Next;
} QueueNode;

typedef struct
{
    QueuePointer Front;
    QueuePointer Rear;
} QueueType;

typedef enum {
    FALSE, TRUE
} boolean;


void CreateQ(QueueType *Queue);
boolean EmptyQ(QueueType Queue);
void AddQ(QueueType *Queue, QueueElementType Item);
void RemoveQ(QueueType *Queue, QueueElementType *Item);

/*??????? ?????? ???? ????????? AddQ ???? ?? ????? ??????? ? ????.*/
int main()
{

    /*?????? ??????????.*/
    QueueType AQueue;
    QueuePointer TempPtr;
    int i,N,M;

    /*?????????? ?????.*/
    CreateQ(&AQueue);

    /*??????? ??????????? ??? ???????? ??????? ???????????.*/
    do{
        printf("DWSE TO PLITHOS TWN FYLAKISMENWN:");
        scanf("%d",&N);

        if(N < 1 )
            printf("Try again,the number must me >=1.\n");

    }while(N < 1);

    /*??????? ???? ???? ???????? ???? ??? ?? ?????? ??? ???????????.*/
    for(i=1; i <= N; i++)
        AddQ(&AQueue,i);

    /*??????? ??????????? ??? ???????? ??????? ?????????.*/
    do{
        printf("DWSE TON ARITHMO TOU KYKLOU EKTELESIS:");
        scanf("%d",&M);

        if(M < 1 )
            printf("Try again,the number must me >=1.\n");

    }while(M < 1);


```

```
    /*???? ?????????? ????? ?? ?????? ???? ???? ???????????.*/
    while(AQueue.Front != AQueue.Front->Next)
    {

        for(i=1; i < M; i++)
        {
            printf("%d, ",AQueue.Front->Data);

            /*?? ????? ? ???????????? ?????????? (??????? ?? ??? ?????? ??? ?????? ?????????)
            ???? ??????? ?? Next ??? ?? ??????? ??? ?????????? ????????,????? ?? ??????? ??
            "??????????".
            ?? ??? ????? ?? ???????????? ,???? ???????? ??? ???????.*/

            if(i == M-1)
            {
            TempPtr = AQueue.Front;
            AQueue.Front = AQueue.Front->Next;
            TempPtr->Next = AQueue.Front->Next;
            }
            else
            {
               AQueue.Front = AQueue.Front->Next;
            }
        }

        /*???????? ??? ?????????? ????????????, ? ?????? ?? ?????????? ??????.*/
        printf("%d, ",AQueue.Front->Data);
        printf("EXECUTION %d\n",AQueue.Front->Data);

        /*????????? ???? ??????? ??????????.*/
        AQueue.Front = AQueue.Front->Next;


    }
    /*???????? ??? ??????????? ??? ??????.*/
    printf("SURVIVAL %d",AQueue.Front->Data);

    return 0;
}


void CreateQ(QueueType *Queue)
{
    Queue->Front = NULL;
    Queue->Rear = NULL;
}

boolean EmptyQ(QueueType Queue)
{
    return (Queue.Front==NULL);
}

void AddQ(QueueType *Queue, QueueElementType Item)
{
    QueuePointer TempPtr;

    TempPtr= (QueuePointer)malloc(sizeof(struct QueueNode));
    TempPtr->Data = Item;
    /*?? NEXT ??? ?????????? ?????? ?? ?????? ???? ????? ?????.*/
    TempPtr->Next = Queue->Front;
    if (Queue->Front==NULL)
    {
        Queue->Front=TempPtr;
    }
    else
        Queue->Rear->Next = TempPtr;
```

```
133      Queue->Rear=TempPtr;
134  }
135
136  void RemoveQ(QueueType *Queue, QueueElementType *Item)
137  {
138      QueuePointer TempPtr;
139
140      if (EmptyQ(*Queue)) {
141          printf("EMPTY Queue\n");
142      }
143     else
144     {
145          TempPtr = Queue->Front;
146          *Item=TempPtr->Data;
147          Queue->Front = Queue->Front->Next;
148          free(TempPtr);
149          if (Queue->Front==NULL) Queue->Rear=NULL;
150      }
151  }
152
153
154
```