

【示例-1】 若用一个大小为6的数组来实现循环队列，队头指针**front**指向队列中队头元素的前一个位置，队尾指针**rear**指向队尾元素的位置。若当前**rear**和**front**的值分别为0和3，当从队列中删除一个元素，再加入两个元素后，**rear**和**front**的值分别为（ ）。

A. 1和5 B. 2和4 C. 4和2 D. 5和1

■ $\text{rear}=0, \text{front}=3, \text{MaxSize}=6$

■ $\text{front}=(\text{front}+1)\% \text{MaxSize}=4$

■ $\text{rear}=(\text{rear}+2)\% \text{MaxSize}=2$



B

【示例-2】 对于循环队列，写出求队列中元素个数的公式，并编写相应的算法。

解：循环队列元素个数的计算公式如下：

队中元素个数= $(rear-front+MaxSize)\%MaxSize$

对应的算法如下：

```
int QueueLength(SqQueue Q)
{
    // 返回Q的元素个数，即队列的长度
    return (Q.rear - Q.front + MAXQSIZE) % MAXQSIZE;
}
```

【示例-3】 如果用一个大小为MaxSize的环形数组表示队列，该队列只有一个队头指针front，不设队尾指针rear，而改置一个计数器count，用以记录队列中的元素个数。

- (1) 队列中最多能容纳多少个元素？
- (2) 设计实现队列基本运算的算法。

解：依题意，设计队列的类型如下：

```
// -----静态存储结构-----  
typedef struct  
{  
    ElemType data[MaxSize]; //存放队列中的元素  
    int front;               //队头指针  
    int count;               //队列中元素个数  
} SqQueue;
```

(1) 队列中最多可容纳MaxSize个元素，因为这里不需要空出一个位置以区分队列空和队列满的情况。

(2) 队列sq为空的条件是： `sq.count==0`;

队列为满的条件是： `sq.count==MaxSize`;

队尾元素的位置是：

`(sq.front+sq.count)%MaxSize`。

//-----队初始化算法-----

void InitQueue(SqQueue &qu)

{

 qu.front=qu.count=0;

}

//-----销毁队列算法-----

void DestroyQueue(SqQueue sq)

{ }

//----入队列算法----

```
int EnQueue(SqQueue &sq, ElemType x)
{
    if(sq.count==MaxSize)    //队满
        return 0;
    sq.data[(sq.front+sq.count)%MaxSize]=x;
    sq.count++;              //队中元素个数增1
    return 1;
}
```


//----出队列算法----

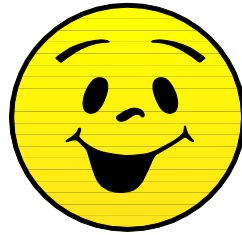
```
int DeQueue(SqQueue &sq, ElemType &x)
{
    if (sq.count==0)           //队空
        return 0;
    x=sq.data[sq.front];
    sq.front=(sq.front+1)%MaxSize;
    sq.count--;                //队中元素个数减1
    return 1;
}
```

//----取队头元素算法----

```
int GetHead(SqQueue sq, ElemType &x)
{
    if(sq.count==0)    //队空
        return 0;
    x=sq.data[sq.front];
    return 1;
}
```

//----判队空算法----

```
int QueueEmpty(SqQueue sq)
{
    if (sq.count==0)
        return 1;           //队空返回1
    else
        return 0;           //队不空返回0
}
```



— END —