

本节内容

# 队列

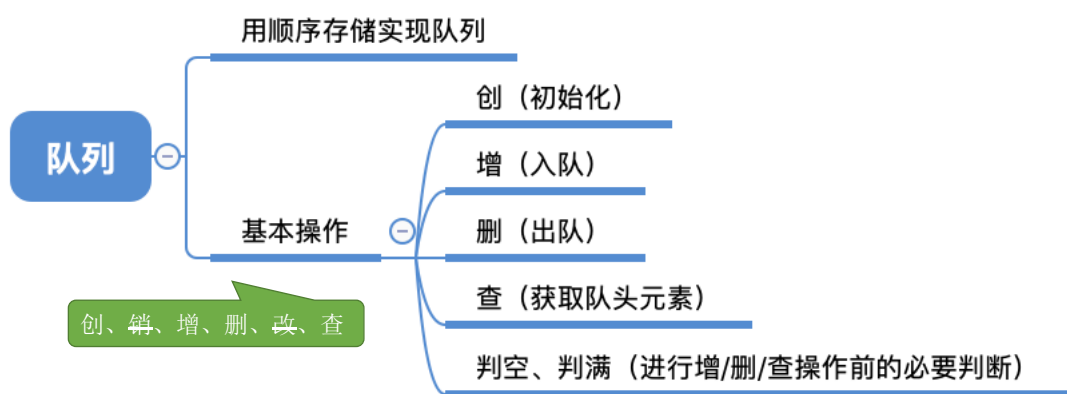
## 顺序实现

王道考研/CSKAOYAN.COM

1

公众号： 考研发条      一手课程！

### 知识总览



王道考研/CSKAOYAN.COM

2

### 队列的顺序实现

```

#define MaxSize 10
typedef struct{
    ElemType data[MaxSize];
    int front, rear;
} SqQueue;

void testQueue(){
    SqQueue Q;
}

```

**Sq: sequence — 顺序**

**rear**  
英 [riə(r)] 美 [riːr]  
n. 后面; 后方部队; 屁股  
adj. 后方的, 后面的; 背面的

**front**  
英 [frʌnt] 美 [frʌnt]  
n. 前面; 正面; 前线  
vt. 面对; 朝向; 对付  
vi. 朝向  
adj. 前面的; 正面的

//定义队列中元素的最大个数

//用静态数组存放队列元素

//队头指针和队尾指针

连续的存储空间, 大小 MaxSize\*sizeof(ElemType)

指向队尾元素的后一个位置 (下一个应该插入的位置)

指向队头元素

内存

0	front
5	rear
	data[9]
	data[8]
	data[7]
	data[6]
	data[5]
	e
	d
	c
	b
	a

王道考研/CSKAOYAN.COM

3

公众号： 考研发条      一手课程！

### 初始化操作

```

#define MaxSize 10
typedef struct{
    ElemType data[MaxSize];
    int front, rear;
} SqQueue;

//初始化队列
void InitQueue(SqQueue &Q){
    //初始时 队头、队尾指针指向0
    Q.rear=Q.front=0;
}

void testQueue(){
    //声明一个队列 (顺序存储)
    SqQueue Q;
    InitQueue(Q);
}

```

**增删改查**

//定义队列中元素的最大个数

//用静态数组存放队列元素

//队头指针和队尾指针

//判断队列是否为空

```

bool QueueEmpty(SqQueue Q){
    if(Q.rear==Q.front) //队空条件
        return true;
    else
        return false;
}

```

内存

0	front
1	rear
	data[9]
	data[8]
	data[7]
	data[6]
	data[5]
	data[4]
	data[3]
	data[2]
	data[1]
	a

王道考研/CSKAOYAN.COM

4

**入队操作** 只能从队尾入队（插入）

```

#define MaxSize 10           //定义队列中元素的最大个数
typedef struct{
    ElemType data[MaxSize];  //用静态数组存放队列元素
    int front, rear;         //队头指针和队尾指针
} SqQueue;

//入队
bool EnQueue(SqQueue &Q, ElemType x){
    if(队列已满)            //队满则报错
        return false;
    Q.data[Q.rear]=x;        //将x插入队尾
    Q.rear=Q.rear+1;         //队尾指针后移
    return true;
}

```

队列已满的条件：  
rear==MaxSize ???

内存

王道考研/CSKAOYAN.COM

5

公众号： 考研发条

一手课程！

**入队操作** 只能从队尾入队（插入）

```

#define MaxSize 10           //定义队列中元素的最大个数
typedef struct{
    ElemType data[MaxSize];  //用静态数组存放队列元素
    int front, rear;         //队头指针和队尾指针
} SqQueue;

//入队
bool EnQueue(SqQueue &Q, ElemType x){
    if(队列已满)            //队满则报错
        return false;
    Q.data[Q.rear]=x;        //新元素插入队尾
    Q.rear=(Q.rear+1)%MaxSize; //队尾指针加1取模
    return true;
}

```

模运算将无限的整数域映射到有限的整数集合  $\{0, 1, 2, \dots, b-1\}$  上

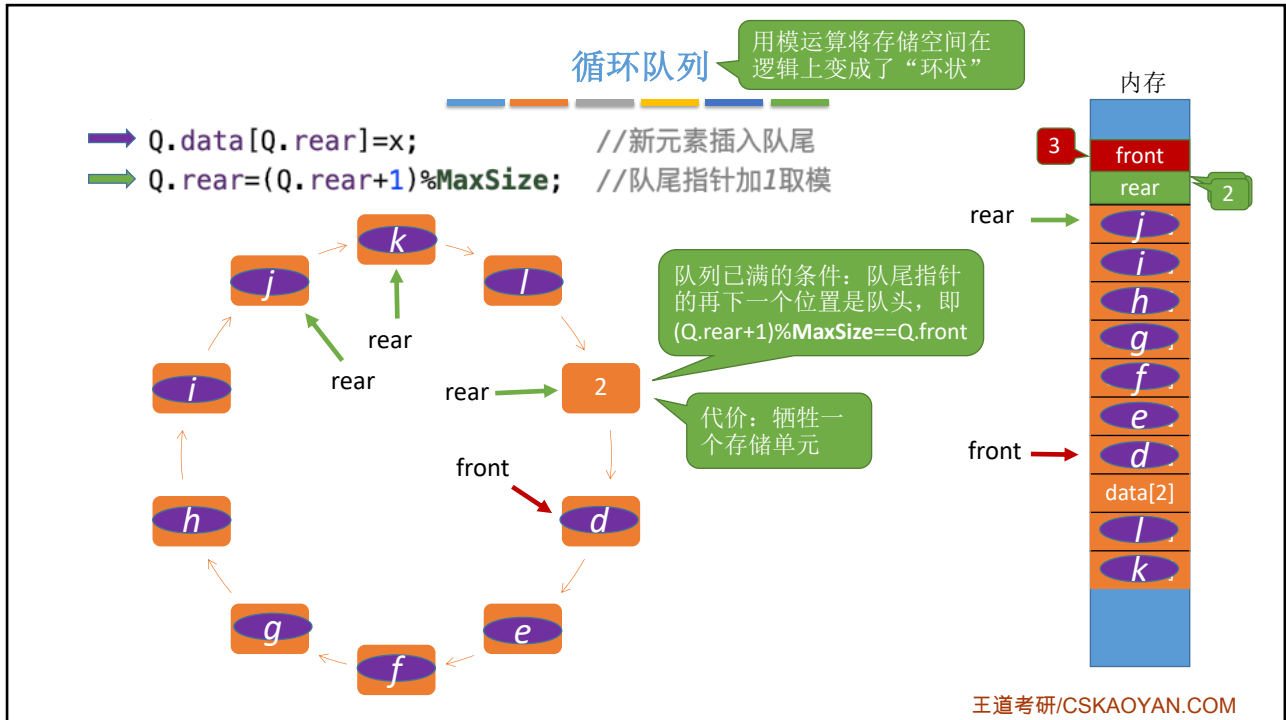
$\{0, 1, 2, \dots, \text{MaxSize}-1\}$  将存储空间在逻辑上变成了“环状”

跨考Tips: 取模运算, 即取余运算。两个整数  $a, b$ ,  $a \% b == a$  除以  $b$  的余数  
在《数论》中, 通常表示为  $a \text{ MOD } b$

内存

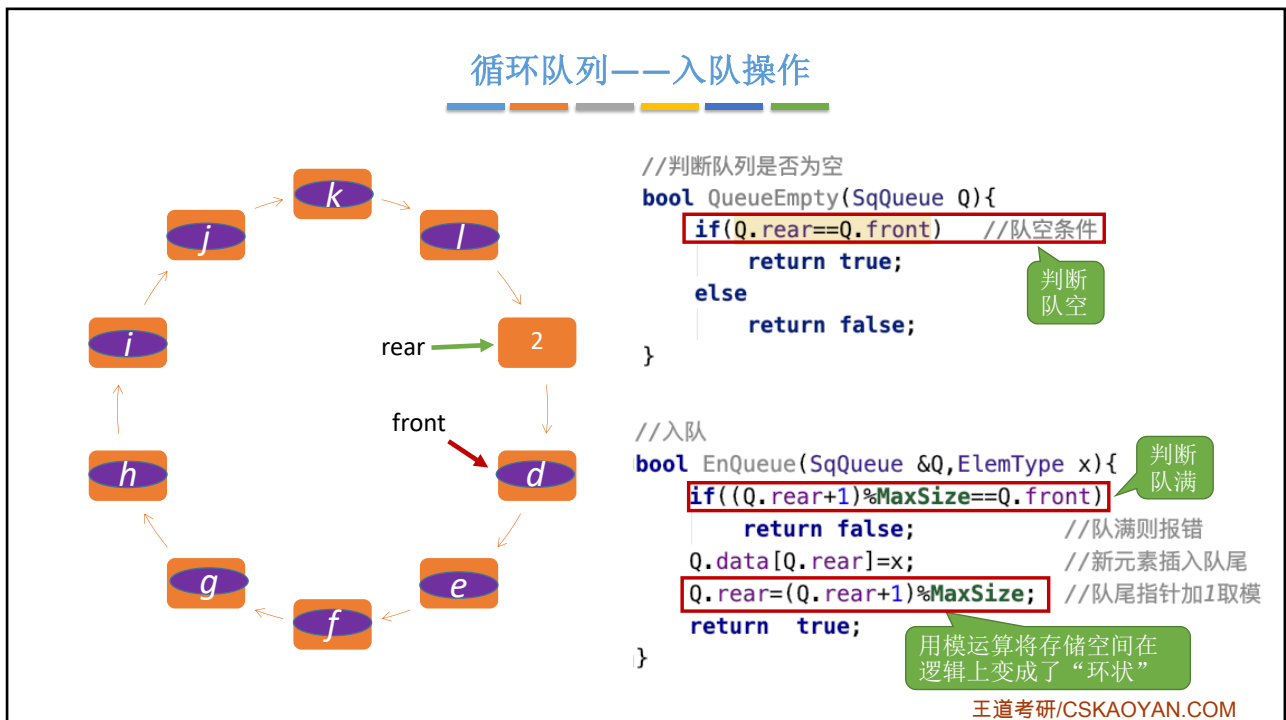
王道考研/CSKAOYAN.COM

6



7

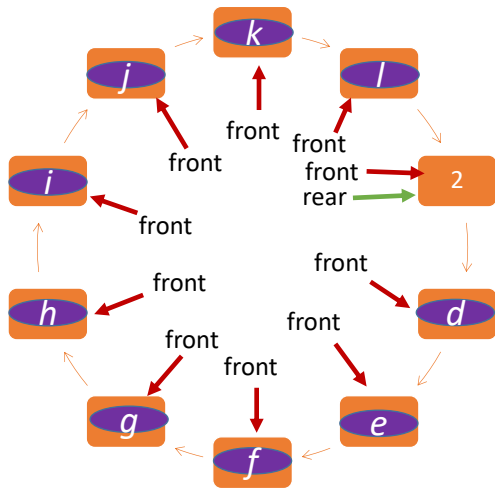
公众号： 考研发条      一手课程！



8

## 循环队列——出队操作

只能让队头元素出队



```
//出队 (删除一个队头元素, 并用x返回)
bool DeQueue(SqQueue &Q, ElemType &x){
    if(Q.rear==Q.front) //判断队空
        return false; //队空则报错
    x=Q.data[Q.front];
    Q.front=(Q.front+1)%MaxSize; //队头指针后移
    return true;
}
```

```
//获得队头元素的值, 用x返回
bool GetHead(SqQueue Q, ElemType &x){
    if(Q.rear==Q.front)
        return false; //队空则报错
    x=Q.data[Q.front];
    return true;
}
```

王道考研/CSKAOYAN.COM

9

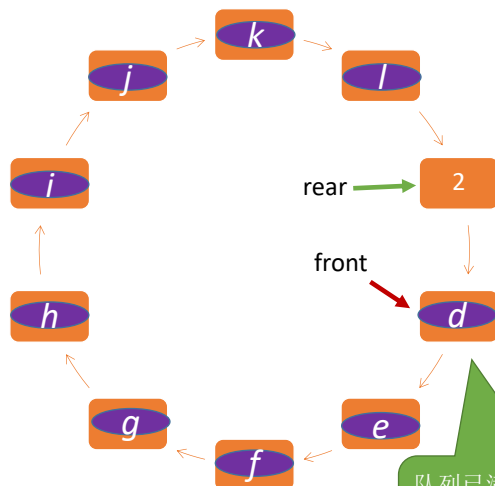
公众号： 考研发条

一手课程！

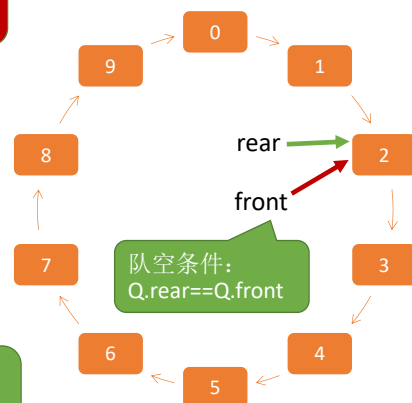
队列元素个数：  
 $(\text{rear} + \text{MaxSize} - \text{front}) \% \text{MaxSize}$

## 方案一：判断队列已满/已空

```
#define MaxSize 10
typedef struct{
    ElemType data[MaxSize];
    int front, rear;
} SqQueue;
```

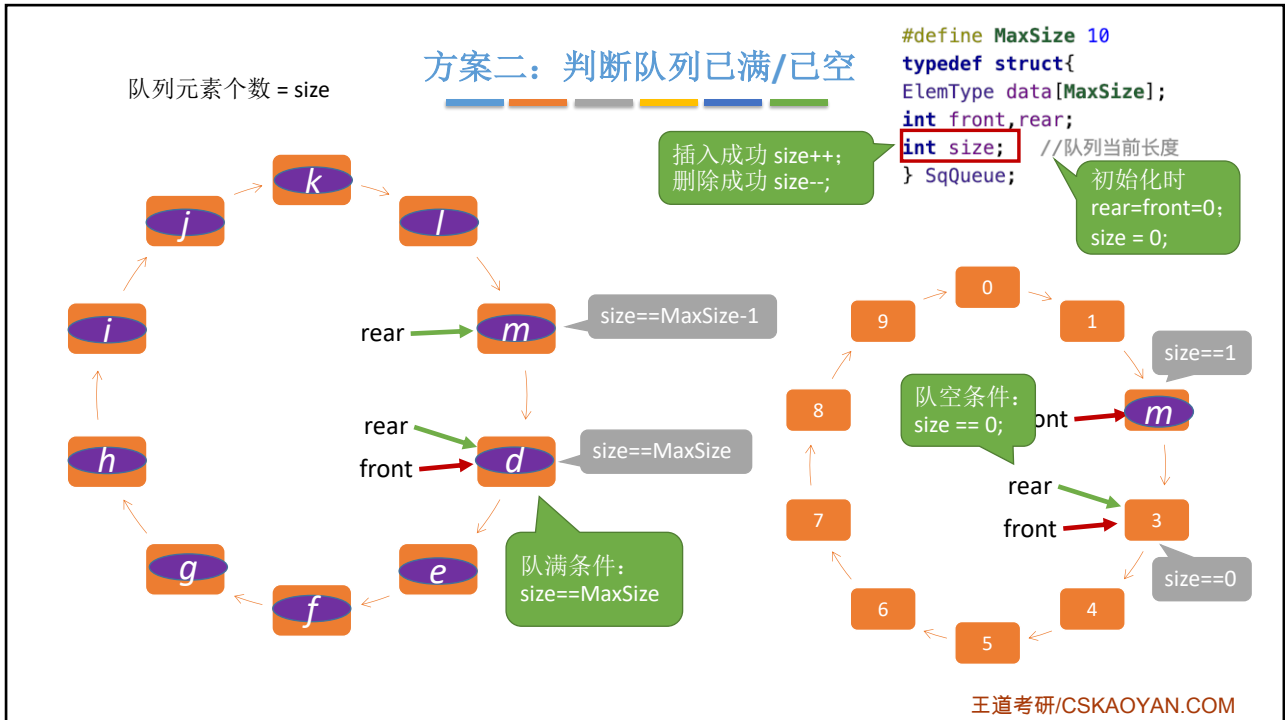
初始化时  
rear=front=0出题老师：不准  
浪费这可怜孤独的  
存储空间!!!

队列已满的条件：队尾指针  
的再下一个位置是队头，即  
 $(\text{Q.rear} + 1) \% \text{MaxSize} == \text{Q.front}$

队空条件：  
Q.rear==Q.front

王道考研/CSKAOYAN.COM

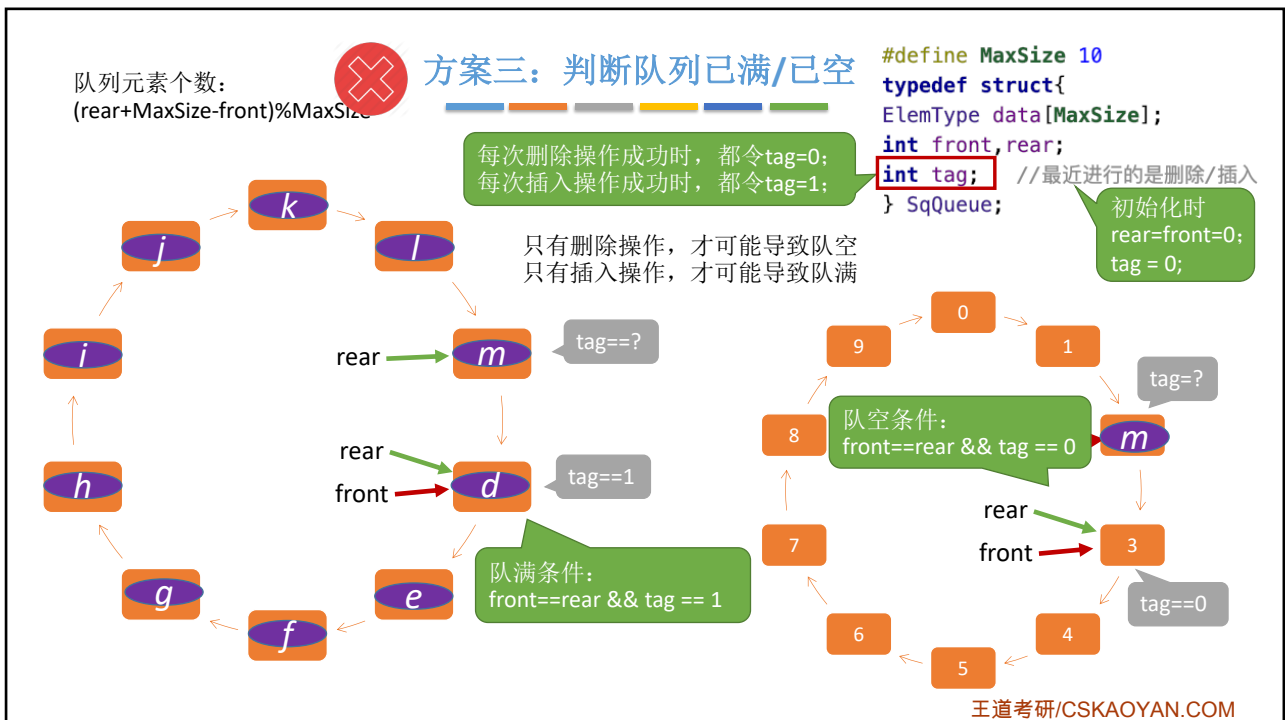
10



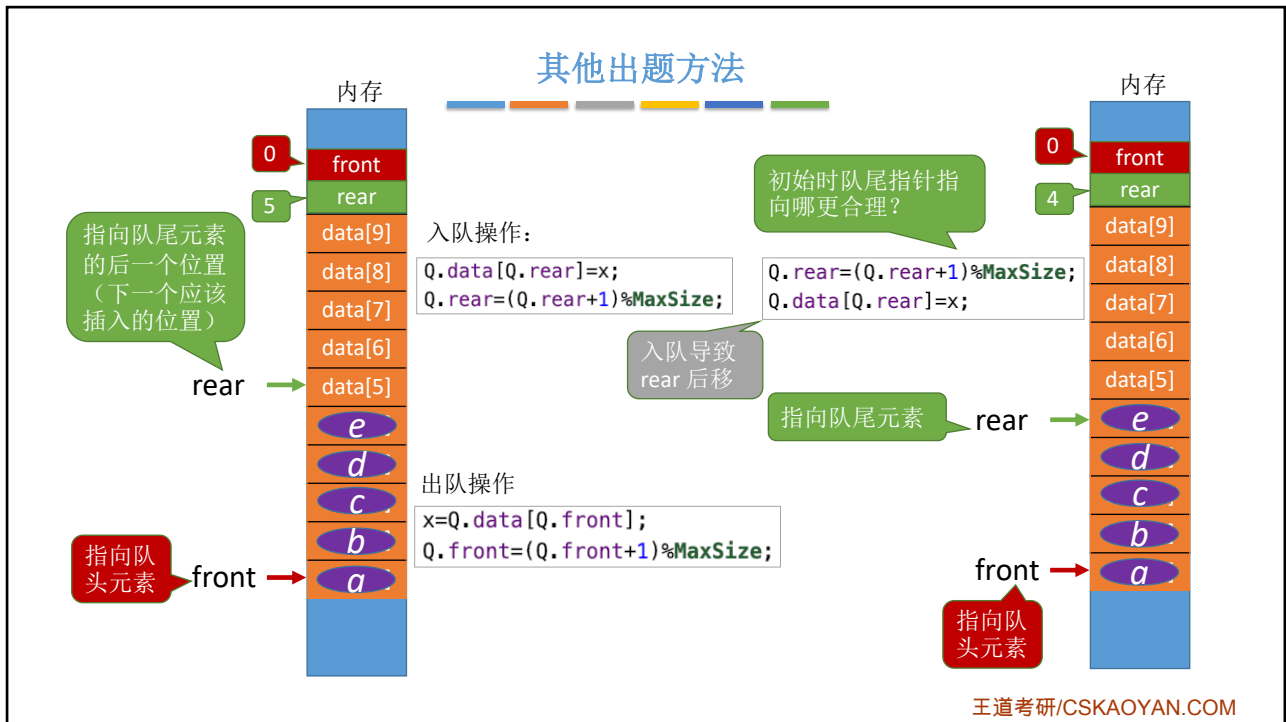
11

公众号： 考研发条

一手课程！



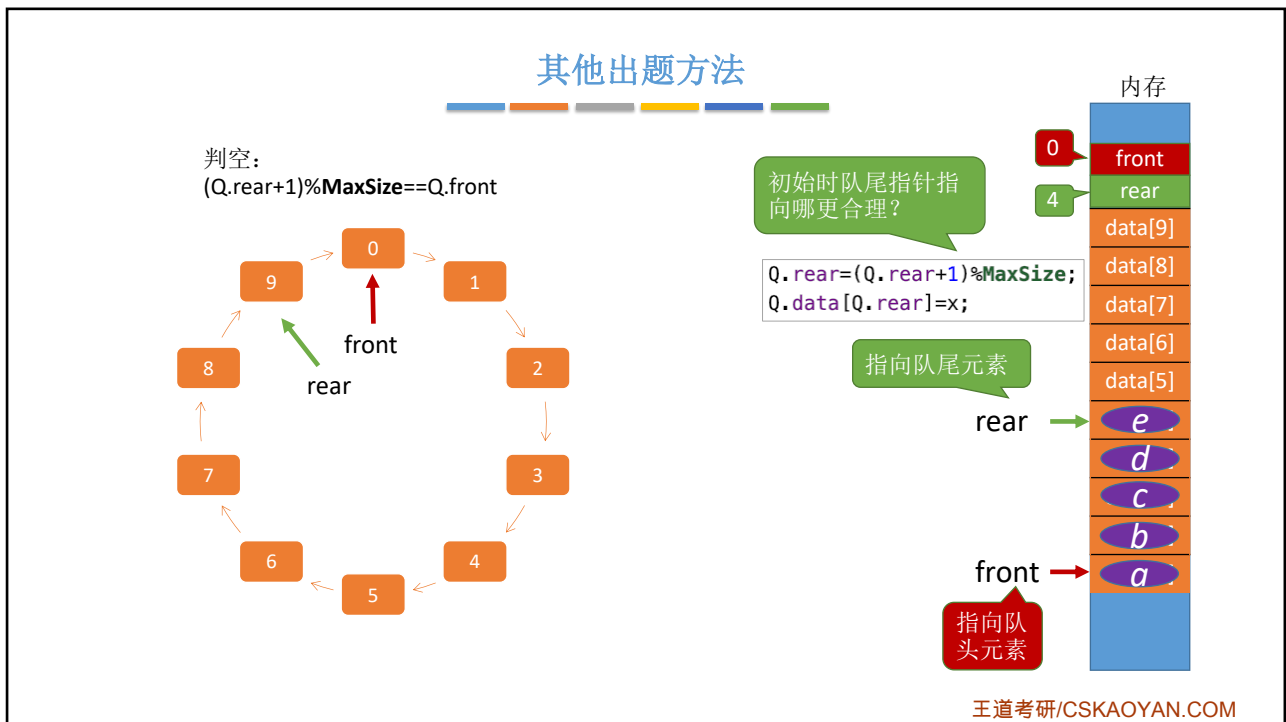
12



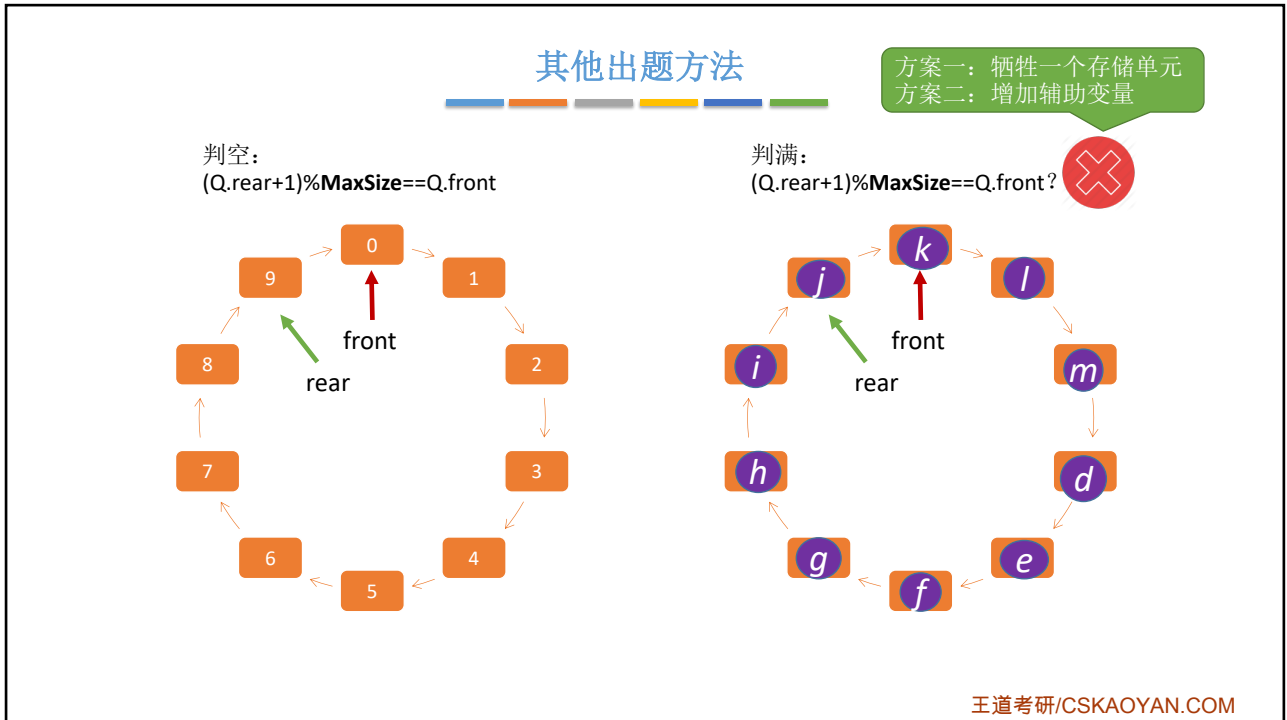
13

公众号： 考研发条

一手课程！

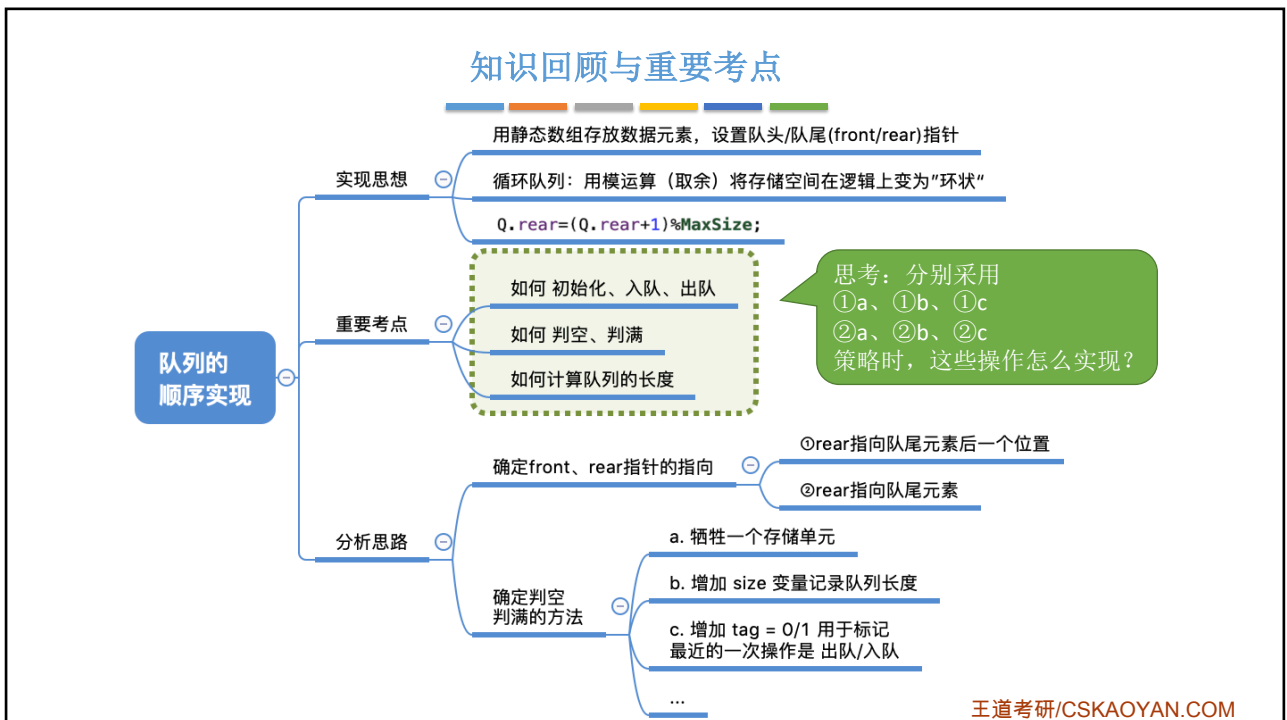


14



15

公众号： 考研发条      一手课程！



16