本小节内容

2019 年 42 题真题讲解

2019 年 42 题真题讲解

2019 年

42. （10 分）请设计一个队列，要求满足：①初始时队列为空；②入队时，允许增加队列占用空间；③出队后，出队元素所占用的空间可重复使用，即整个队列所占用的空间只增不减；④入队操作和出队操作的时间复杂度始终保持为 *O*(1)。请回答下列问题：

（1）该队列是应选择链式存储结构，还是应选择顺序存储结构？

（2）画出队列的初始状态，并给出判断队空和队满的条件。

（3）画出第一个元素入队后的队列状态。

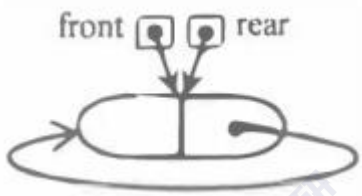
（4）给出入队操作和出队操作的基本过程。

答案解析：

(1) 采用链式存储结构(两段式单向循环链表)，队头指针为 front，队尾指针为 rear。

因为第二个要求入队时，允许增加队列占用空间，所以必须使用链式存储

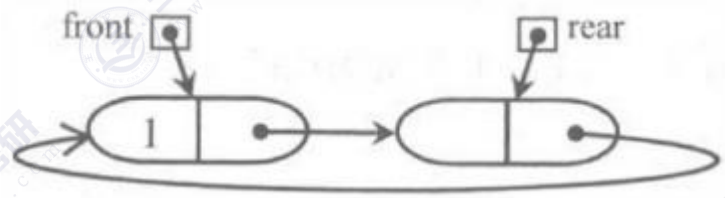
(2) 初始时，创建只有一个空闲结点的两段式单向循环 链表，头指针 front 与尾指针 rear 均指向空闲结点。如下图所示。



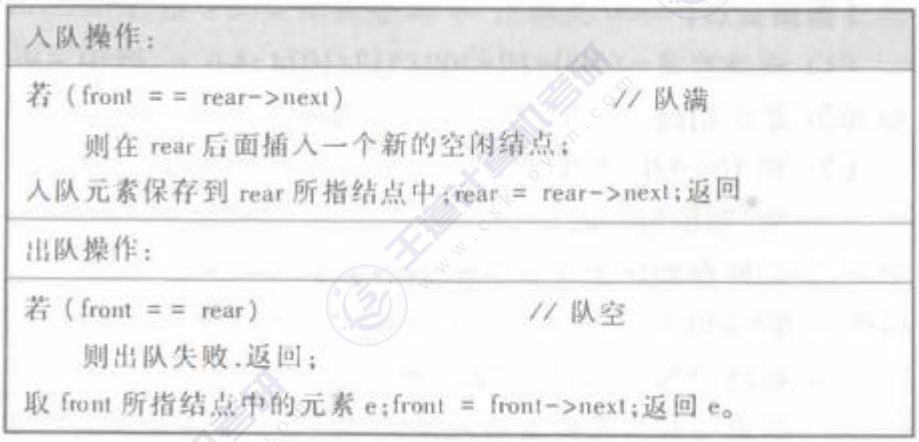
队空的判定条件：front==rear。

队满的判定条件：front==rear->next。

(3)插入第一个元素后的队列状态：



（4）入队操作和出队操作的基本过程



这道题目考研并没有要求大家实现代码，但是为了大家更清晰的理解原理，源码如下：

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

typedef int ElemType;
typedef struct LNode {
	ElemType data;
	struct LNode* next;//指向下一个结点
}LNode, * LinkList;


void EnQueue(LinkList front, LinkList& rear, ElemType val)
{
	LinkList pnew;
	if (rear->next == front)
	{
		//队列满，申请一个结点的空间，放入队列
		pnew= (LinkList)malloc(sizeof(LNode));
		rear->data = val;//把入队元素放入 rear 指向结点
		rear->next = pnew;//放了一个结点，其相当于做了分割
		pnew->next = front;
		rear = pnew;
	}
	else {//如果队列不满，直接放值，让 rear 后移一个结点
		rear->data = val;
		rear = rear->next;
	}
}

void DeQueue(LinkList& front, LinkList rear)
{
	if (front == rear)
	{
```

```
            printf("队列为空\n");
        }
        else {
            printf("出队值为%d\n", front->data);
            front = front->next;
        }
}

void CircleQueue(LinkList& front, LinkList& rear)
{
        //队列头和队列尾都指向一个结点，这时队列既是空的，也是满的
        front = (LinkList)malloc(sizeof(LNode));
        rear = front;
        rear->next = front;
        //入队
        EnQueue(front, rear, 3);
        EnQueue(front, rear, 4);
        //出队
        DeQueue(front, rear);
        DeQueue(front, rear);
        DeQueue(front, rear);
}


int main()
{
        LinkList front, rear;
        CircleQueue(front, rear);
        return 0;
}
```