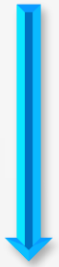
2.3.3 队列的链式存储

队列链表结构分析

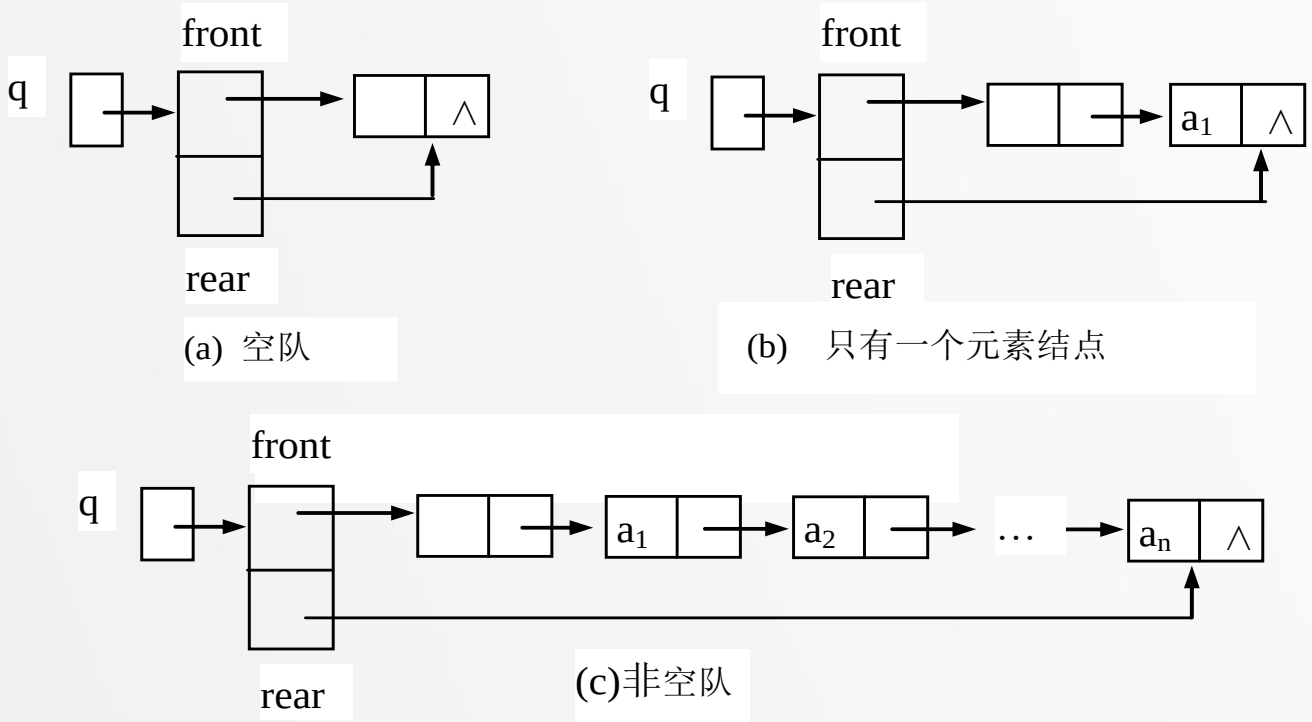
- 主要操作：链表尾部插入，链表头部删除



- 需要有头结点
- 头指针指向第1个结点，尾指针指向最后1个结点

2.3.3 队列的链式存储

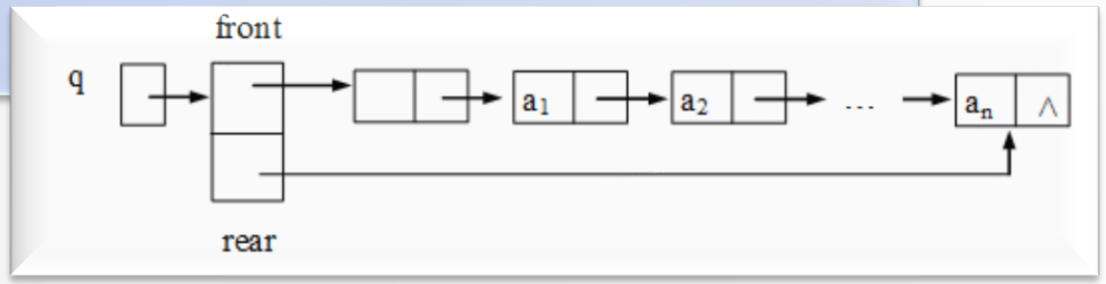
- 2个指针/带头结点的单链表



2.3.3 队列的链式存储

- 类型定义

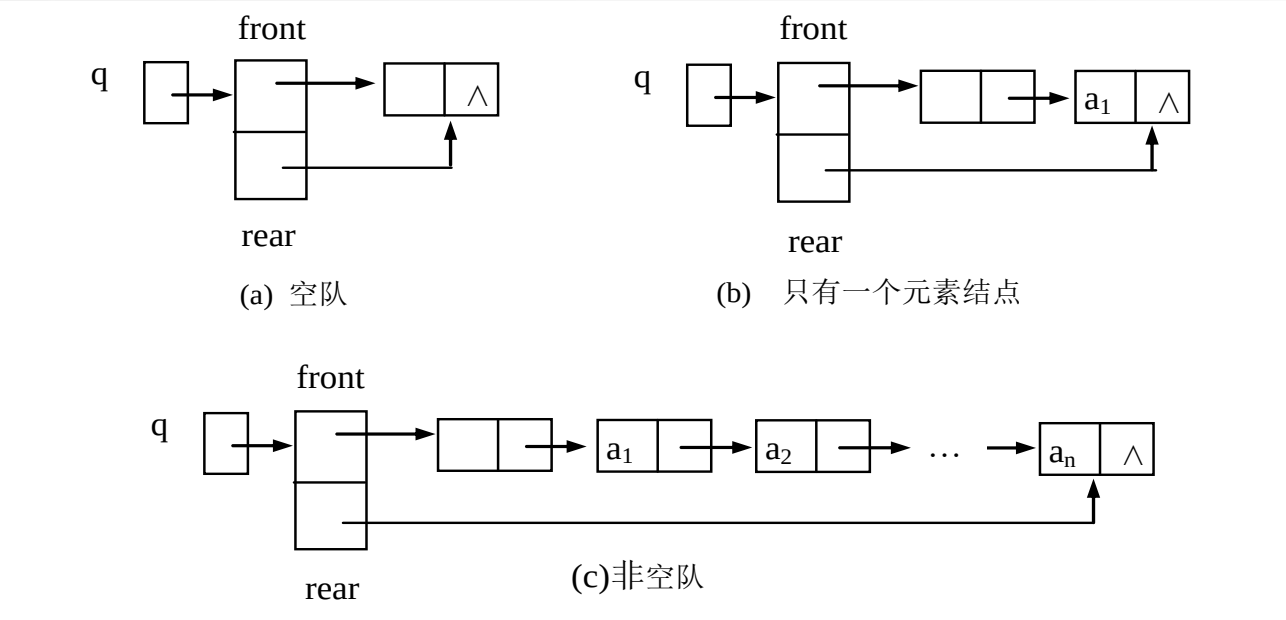
```
typedef struct node {          /*链式队列的结点结构*/
  QueueEntry Entry;          /*队列的数据元素类型*/
  struct node *next;         /*指向后继结点的指针*/
}QueueNode, *QueueNodePtr;
typedef struct queue{          /*链式队列*/
  QueueNode *front;          /*队头指针*/
  QueueNode *rear;           /*队尾指针*/
} Queue,*QueuePtr;
```



2.3.3 队列的链式存储

```
Typedef strcut node{
   ElemType data;
    struct node *next;
}QNode,*QNodePtr;
```

```
typedef struct queue{
   QNode *front,*rear;
}Queue,*QueuePtr;
```



变量定义与使用：

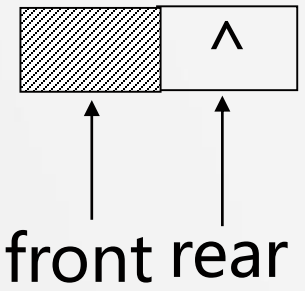
QueuePtr Q;

QNode hNode;

Q->front=& hNode;

Q->rear=& hNode;

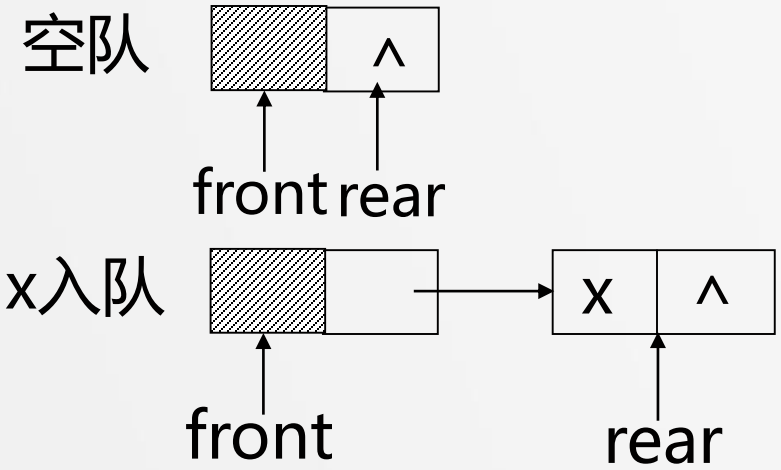
空队



初始化

Q->front->next=NULL

Q->rear->next=Null

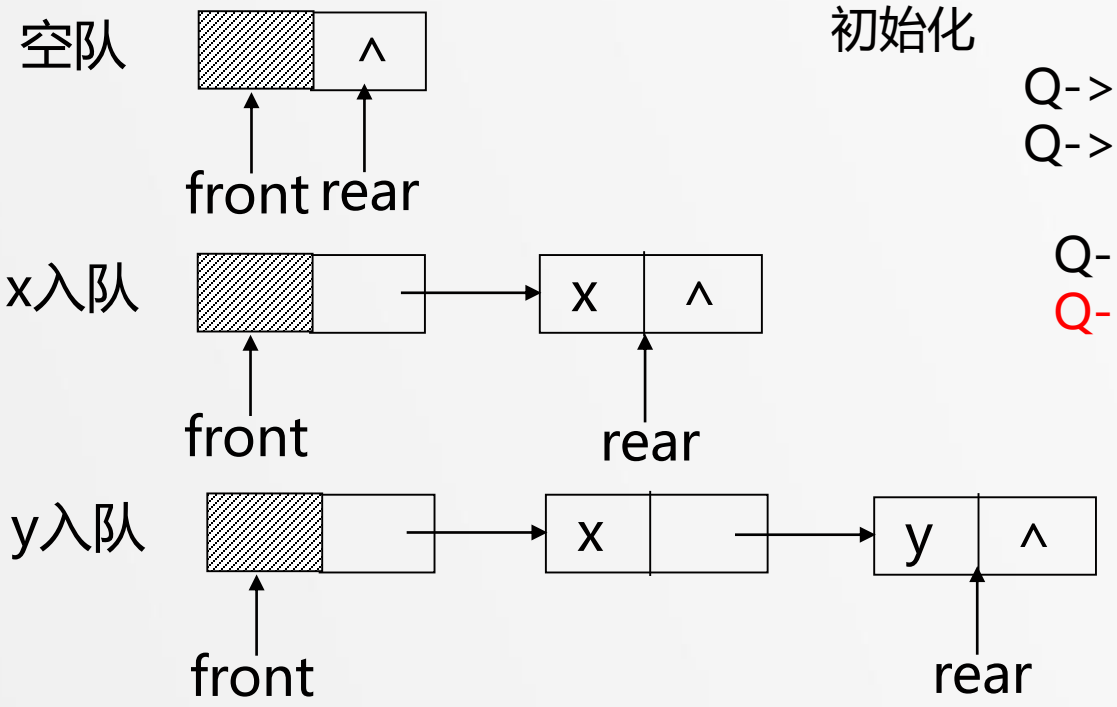


初始化

Q->front->next=NULL
Q->rear->next=Null

px=new Qnode;
px->data=x;
px->next=NULL;

Q->rear->next = px;
Q->rear = px;



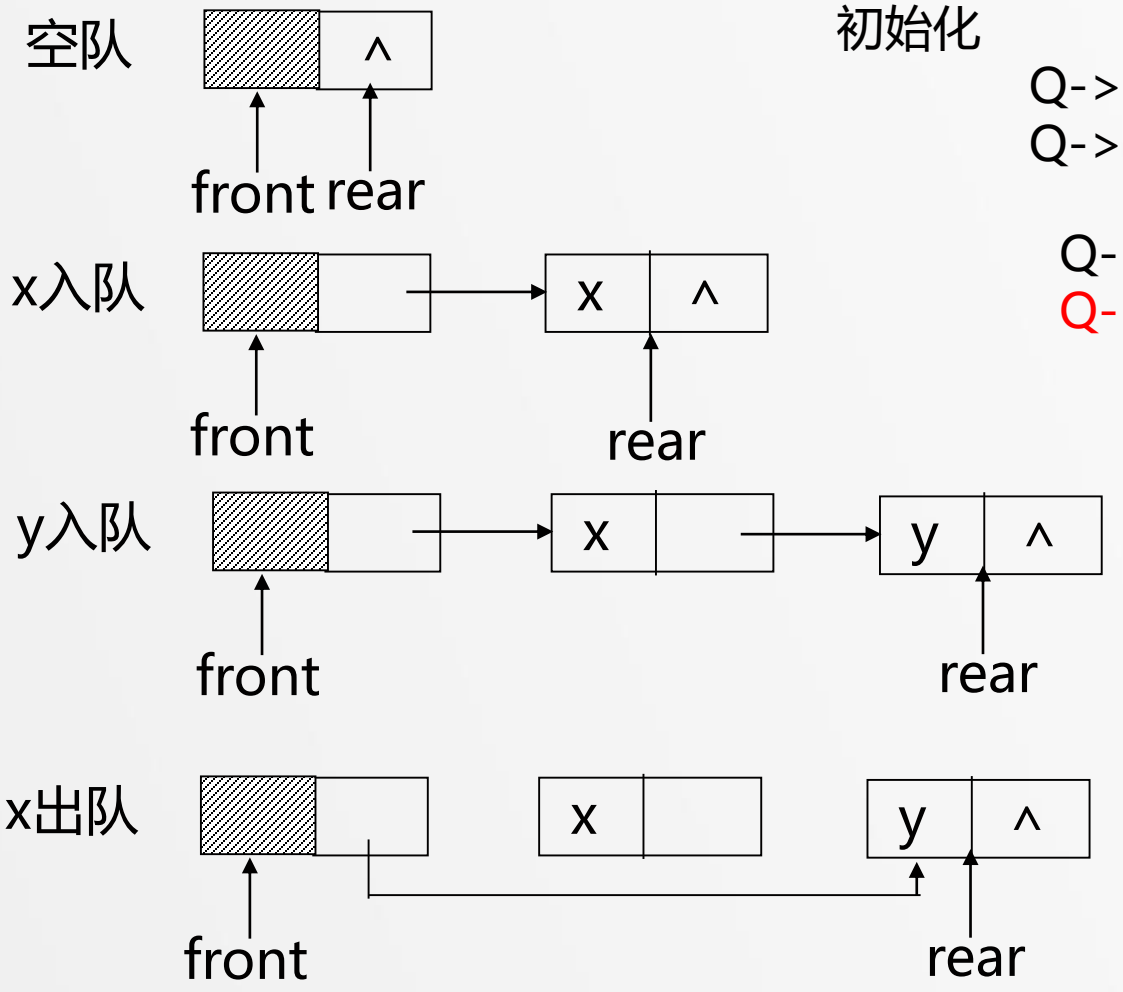
初始化

Q->front->next==NULL
Q->rear->next==Null

Q->rear->next = px;
Q->rear = px;

px=new Qnode;
px->data=y;
px->next=NULL;

Q->rear->next = px;
Q->rear = px;



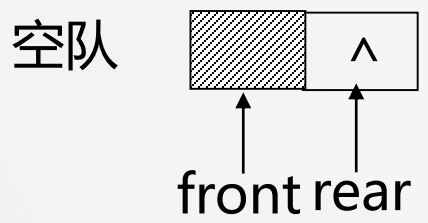
初始化

Q->front->next=NULL
Q->rear->next=Null

Q->rear->next = px;
Q->rear = px;

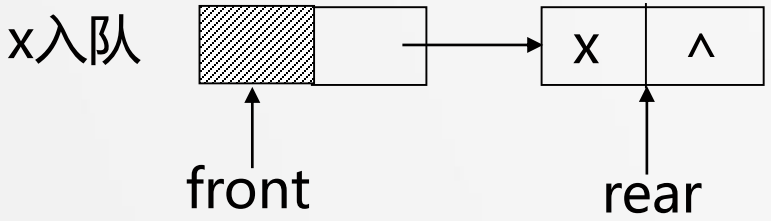
Q->rear->next = px;
Q->rear = px;

px=Q->front->next;
Q->front->next = Q->front->next->next;
free(px);

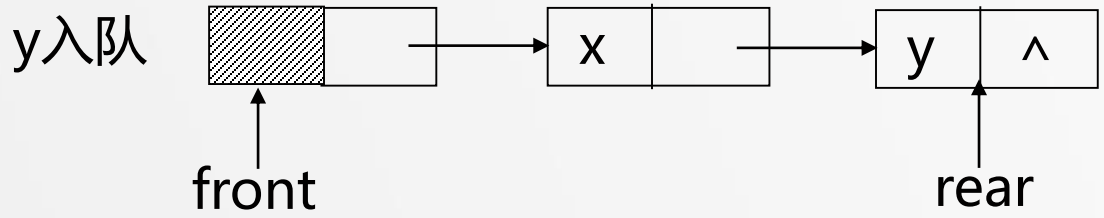


初始化

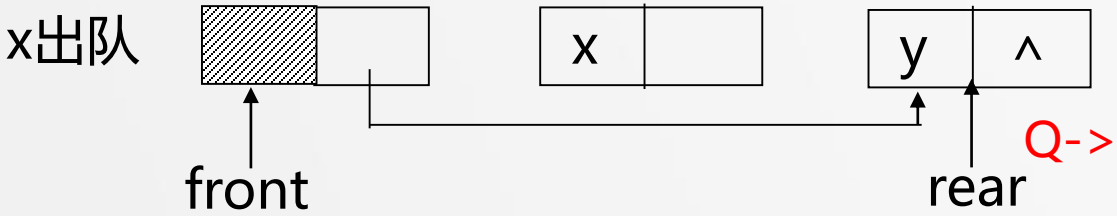
Q->front->next==NULL
Q->rear->next==Null



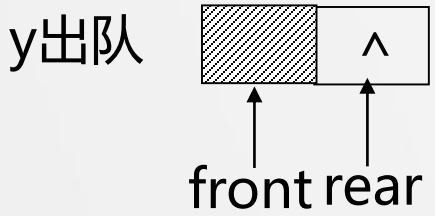
Q->rear->next = px;
Q->rear = px;



Q->rear->next = px;
Q->rear = px;



Q->front->next = Q->front->next->next



Q->front ->next= Q->front->next->next;
Q->rear = Q->front;