**栈和队列**

栈（Stack）又称堆栈，是运算受限的线性表

其限制是仅允许在表的一端进行插入和删除操作，不允许在其他任何位置进行插入、查找、删除等操作

表中进行插入、删除操作的**一端**称为栈顶（top），栈顶保存的元素称为栈顶元素

相对的，表的另一端称为栈底（bottom）

当栈中没有数据元素时称为空栈

向一个栈插入元素又称为进栈或入栈;从一个栈中删除元素又称为出栈或退栈，由于栈的插入和删除操作仅在栈顶进行，后进栈的元素必定先出栈，所以又把堆栈称为后进先出表（Last In First Out,简称LIFO）

栈∶push、pop、peek

**先进去的，后出来**

public interface Stacks{

//返回堆栈的大小

public int getSize();

//判断堆栈是否为空

public boolean isEmpty();

//数据元素 e入栈

public void push(Object e);

//栈顶元素出栈

public Object pop();

//取栈顶元素

public Object peek();

**栈的存储结构**

堆栈也有两种基本的存储结构∶顺序存储结构和链式存储结构

顺序栈：

类似于顺序表

顺序栈是使用顺序存储结构实现的堆栈，即利用一组地址连续的存储单元依次存放堆栈中的数据元素。由于堆栈是一种特殊的线性表，因此在线性表的顺序存储结构的基础上，**选择线性表的一端作为栈顶即可**

根据数组操作的特性，选择数组下标大的一端，即线性表顺序存储的表尾来作为栈顶，此时入栈、出栈等操作可以在O（1）时间完成

由于堆栈的操作都在栈顶完成，因此在顺序栈的实现中需要附设一个指针 top来动态的指示栈顶元素在数组中的位置。通常 top 可以用找顶元素所在数组下标来表示，top=-1 时表示空栈

链栈：

即采用链表作为存储结构实现的栈

当采用单链表存储线性表后，根据单链表的操作特性选择单链表的头部作为栈顶，此时，入栈、出栈等操作可以在O（1）内完成。由于堆栈的操作只在线性表的一端进行，在这里使用带头结点的单链表或不带头结点的单链表都可以。使用带头结点的单链表时，结点的插入和删除都在头结点之后进行;使用不带头结点的单链表时，结点的插入和删除都在链表的首结点上进行

**队列（queue）**

简称**队**，它同堆栈一样，也是一种运算受限的线性表

其限制是仅允许在表的一端进行插入，在表的另一端进行删除

在队列中把**插入**数据元素的一端称为**队尾（rear）**

**删除**数据元素的一端称为**队首（front）**.

向队尾插入元素称为进队或入队，新元素入队后成为新的队尾元素

从队列中删除元素称为离队或出队，元素出队后，其后续元素成为新的队首元素

由于队列的插入和删除操作分别在队尾和队首进行，每个元素必然按照进入的次序离队

也就是说先进队的元素必然先离队，所以称队列为先进先出表（First In FirstOut.简称FIFO）

对于队列的主要操作是**入队**和**出队**操作

public interface Queue{

//返回队列的大小

public int getSize();

//判断队列是否为空

public boolean isEmpty();

//数据元素 e 入队

public void enqueue(Object e);

//队首元素出队

public Object dequeue();

//取队首元素

public Object peek();

队列的存储结构

**顺序队列存储结构**

方法1∶使用数组作为存储结构

方法2∶使用循环数组作为存储结构（不会造成空间浪费）

**链式队列存储结构**

队列的链式存储可以使用单链表来实现

为了操作实现方便，这里采用带头结点的单链表结构

根据单链表的特点，选择链表的头部作为队首，链表的尾部作为队尾

除了链表**头结点**需要通过一个引用来指向之外，还需要一个对链表队尾结点的引用，以方便队列的入队操作的实现

为此一共设置两个指针，一个队首指针和一个队尾指针：

队首指针指向队首元素的前一个结点，即始终指向链表空的头结点

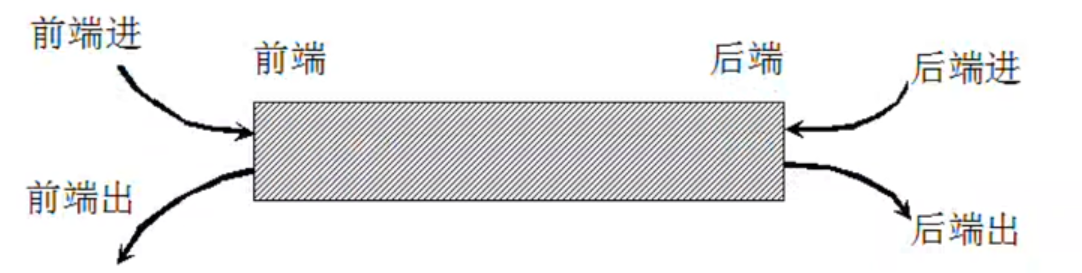
队尾指针指向队列当前队尾元素所在的结点

当队列为空时，队首指针与队尾指针均指向空的头结点

**双端队列（queue double ended queue 通常读为"deck")**

所谓双端队列是指**两端**都可以进行**进队和出队**操作的队列，将队列的两端分别称为**前端和后端**，两端都可以入队和出队

其元素的逻辑结构仍是线性结构



**在双端队列进队时∶**

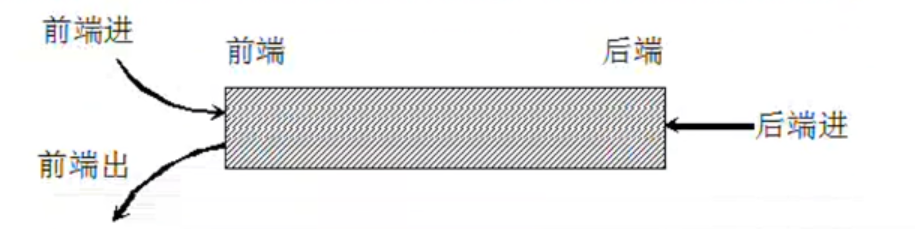
前端进的元素排列在队列中后端进的元素的前面

后端进的元素排列在队列中前端进的元素的后面

在双端队列出队时，无论前端出还是后端出，先出的元素排列在后出的元素的前面

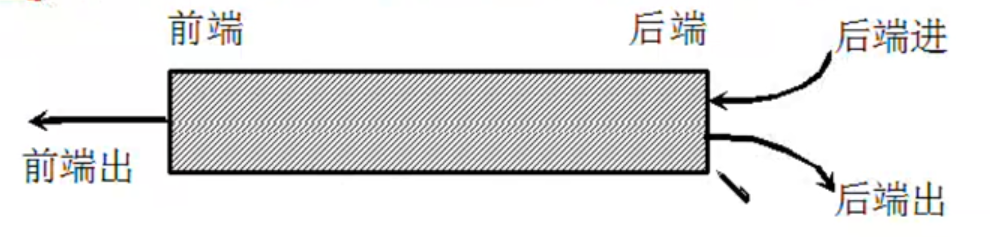
**输出受限的双端队列:**

即一个端点允许插入和删除，另一个端点只允许插入的双端队列



**输入受限的双端队列:**

即一个端点允许插入和删除，另一个端点只允许删除的双端队列



**双端队列既可以用来队列操作，也可以用来实现栈操作（只操作一端就是栈了）**

栈、队列都是线性表

在Java的操作中：Stack类（过时）、Queue类(队列)、Deque类（双端队列）

Queue类是Deque类的父类

Deque类是一个接口类

一般情况下Deque类的实现类是ArrayDeque\LinkedList