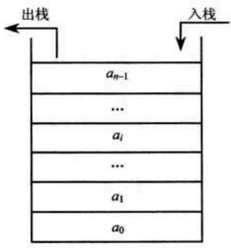
栈和队列

Author: Yi
Date: Dec/29/2015

在上一章中我们涉及了线性表的实现和操作，以及讲解了线性表的顺序存储和链式存储两种方式。这一章中，我们来看看栈和队列。本质上来说，栈和队列也是线性表，只不过添加了一些特殊的功能，也就是我们常说的LIFO, FIFO还有其他一些富有特性的方法。因此在特定情况下，栈和队列的用处也是很广泛的。

1: Stack概念和实现方法

栈是一种特殊的线性表，这种线性表只允许在某一端进行数据操作（插入以及删除），栈通常是栈的尾端。我们通常把进行数据操作，譬如插入和删除的那一端称为栈顶(top)，另一端就称为栈底(bottom)。下图展示了元素入栈和出栈的过程：



可以很清楚的看到，元素入栈顺序和出栈顺序相反，即LIFO(Last in First out).这是栈的一个很重要的性质。
栈常用到的操作有：

- 初始化一个栈
- 元素入栈
- 元素出栈
- 判断栈是否为空
- 清空栈

栈这种线性表既可以用顺序存储来实现，也可以用链式存储来实现。

1.1: 栈的顺序存储实现

因为栈是一种特殊的线性表，但它毕竟还是线性表，因此栈的顺序实现底层也是用一个数组维护的。用arr[size-1] 来访问栈顶元素。栈中元素的逻辑顺序和物理顺序是一致的。

- 元素入栈：判断栈的底层数组是否已经达到最大容量，如果是，对数组进行扩容然后再入栈；如果没有，直接入栈就行。栈长度增加1；
- 元素出栈：判断栈是否为空，如果不为空，返回栈定元素，也即arr[size-1]，栈长度减1；
- 判断栈是否为空：判断长度是否为0.

1.2: 栈的链式存储实现

栈的链式存储和单链表一样，用一个节点来维护链的链接。每个节点包含当前节点的元素和指向下一个节点的地址。栈的链式存储更加简单，只需要维护一个top节点和一个size就好。

- 元素入栈：将新生成的节点的next地址存放原top地址，再更新top地址，长度加1；
- 元素出栈：返回当前top节点值，然后更新top节点为原top的next地址，长度减1；
- 判断栈是否为空： 判断长度是否为0

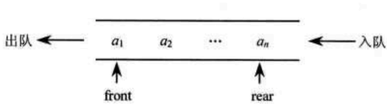
之前我们提过的顺序栈和链式栈的优缺点比较，可以侧面证明链式栈的实现比较高效。

1.3: Java中提供的栈

- Stack: Java中的顺序实现链一个实例类就是Stack。而且是线程安全的。
- LinkedList: Linkedlist是一个双向链表，但是查看Java API看到LinkedList也可以当作一个栈来使用。这是栈的链式实现。LinkedList不是线程安全的。

2: Queue概念和实现方法

Queue也是一种特殊的线性表，和栈不一样的是，队列只能在队尾(rear)插入元素，在队头(front)删除元素。这种先来后到的操作模式又称为FIFIO(First In First Out). 看下图：



队列的常用操作有：

- 初始化一个队列
- 向队列插入元素
- 从队列删除元素
- 判断队列是否为空
- 清空队列

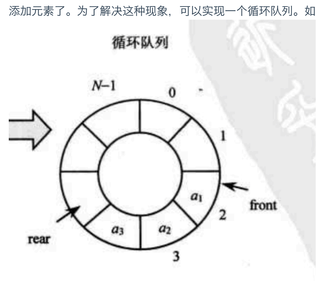
队列是一种特殊的线性表，因此可以像普通线性表那样，可以用顺序存储实现或者链式存储。

2.1: 队列的顺序实现

线性表的顺序实现都是在底层维护一个数组，还有front和rear两个变量。

- 元素入队列： 往队尾插入一个元素，rear增加1，队列长度增加1；
- 元素出队列：从队头删除一个元素，front增加1，队列长度减1；

这种实现和栈并没有太大区别，但是这种实现会出现"假满"现象，也就是说当不断删除元素，front不断增加，最后front和rear总会相等，但此时底层数组并没有存数据，但队列此时已经不能添加元素了。为了解决这种现象，可以实现一个循环队列。如图：



当rear或者front达到数组最大容量时，重置为0；当rear等于front时，如果当前的位置没存储数据，证明队列为空，否则队列满。

2.2: 队列的链式实现

线性表的链式实现都是在底层用一个包含当前节点数据和下一个节点地址的节点实现。同时保留两个节点，分别用来维护front和rear节点。

- 元素入队列： 将新节点链接在原rear节点的后面，同时更新rear和队列长度；
- 元素出队列： 返回当前front节点数据，同时更新front为下一个节点和队列长度；

因为链式存储可以充分利用计算机内存，因此在通常情况下，不会出现队满现象。

2.3: Java中队列的应用

在Java中提供来Queue接口，任意实现了这个接口的实现类都可以当作一个队列来使用。队列常用的方法有：

操作	抛出异常版本	返回特殊值版本
插入	add(e)	offer(e)
删除	remove()	poll()
访问	element()	peek()

2.4: 双向队列

在Java中，除了队列接口Queue之外，还提供了一种双向接口，Deque。双向队列可以看作既实现了Queue接口，同时又实现了Stack接口。因此同时具有队列和栈的性质。Java中实现了ArrayDeque和LinkedList两个实现类，而且LinkedList是线程安全的。因此如果需要栈，通常建议使用LinkedList. 这里是LinkedList的API，可以看到里面既有offer方法，也有push方法，既有poll方法，也有pop方法。