

栈和队列

本节目标

- 学习栈的原理及基本实现
- 学习队列的原理及基本实现
- 熟练使用 java 中的栈和队列

1. 栈(Stack)

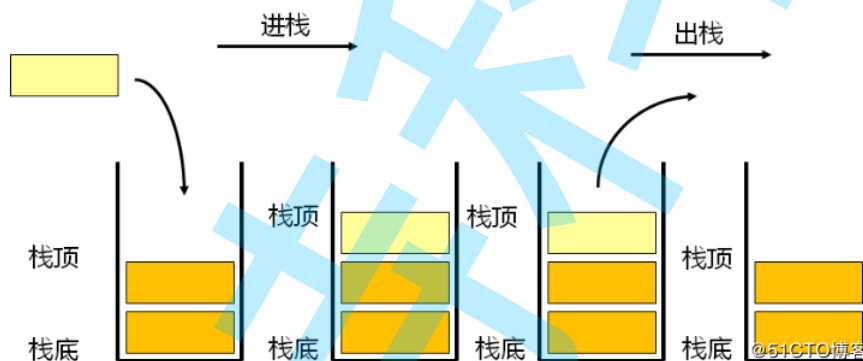
1.1 概念

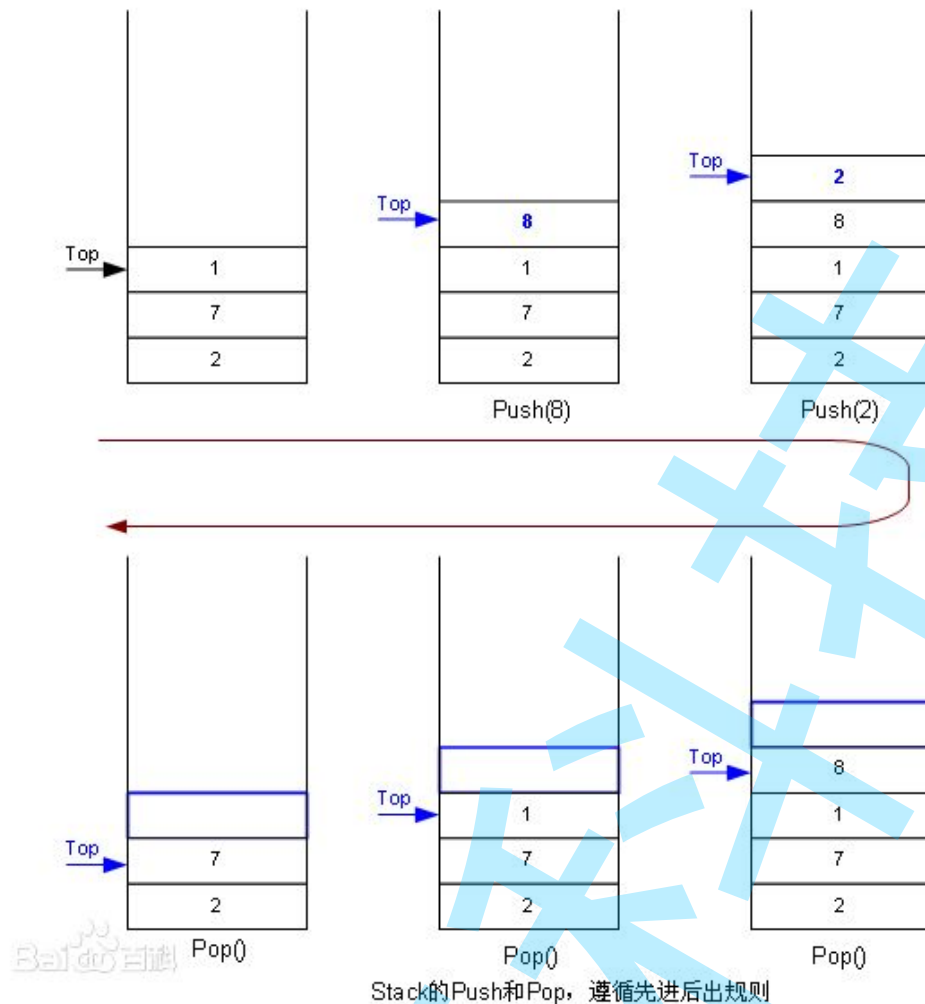
栈：一种特殊的线性表，其只允许在固定的一端进行插入和删除元素操作。进行数据插入和删除操作的一端称为栈顶，另一端称为栈底。栈中的数据元素遵守后进先出LIFO（Last In First Out）的原则。

压栈：栈的插入操作叫做进栈/压栈/入栈，**入数据在栈顶。**

出栈：栈的删除操作叫做出栈。**出数据在栈顶。**

– 后进先出 (Last In First Out)





1.2 实现

1. 利用顺序表实现，即使用尾插 + 尾删的方式实现
2. 利用链表实现，则头尾皆可

相对来说，顺序表的实现上要更为简单一些，所以我们优先用顺序表实现栈。

```

1 public class Mystack {
2     // 简单起见，我们就不考虑扩容问题了
3     private int[] array = new int[100];
4     private int size = 0;
5
6     public void push(int v) {
7         array[size++] = v;
8     }
9
10    public int pop() {
11        return array[--size];
12    }
13
14    public int peek() {
15        return array[size - 1];
16    }
17
18    public boolean isEmpty() {
19        return size == 0;
20    }

```

```

21
22     public int size() {
23         return size;
24     }
25 }

```

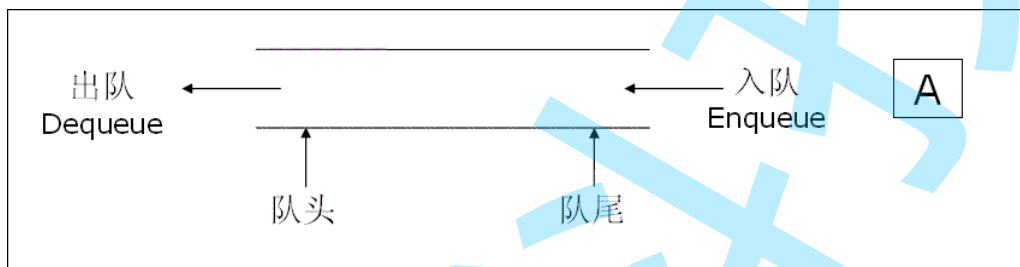
2. 队列(Queue)

2.1 概念

队列：只允许在一端进行插入数据操作，在另一端进行删除数据操作的特殊线性表，队列具有先进先出FIFO(First In First Out)

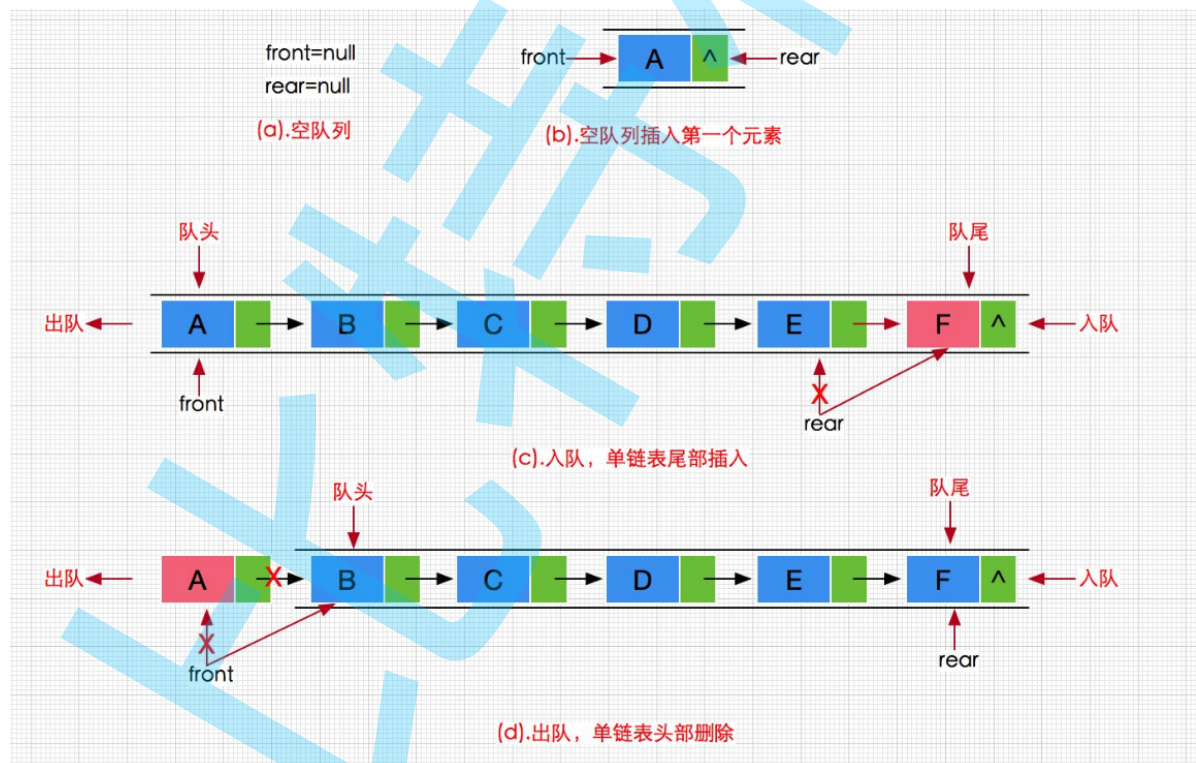
入队列：进行插入操作的一端称为**队尾 (Tail/Rear)**

出队列：进行删除操作的一端称为**队头 (Head/Front)**



2.2 实现

队列也可以数组和链表的结构实现，使用链表的结构实现更优一些，因为如果使用数组的结构，出队列在数组头上出数据，效率会比较低。



```

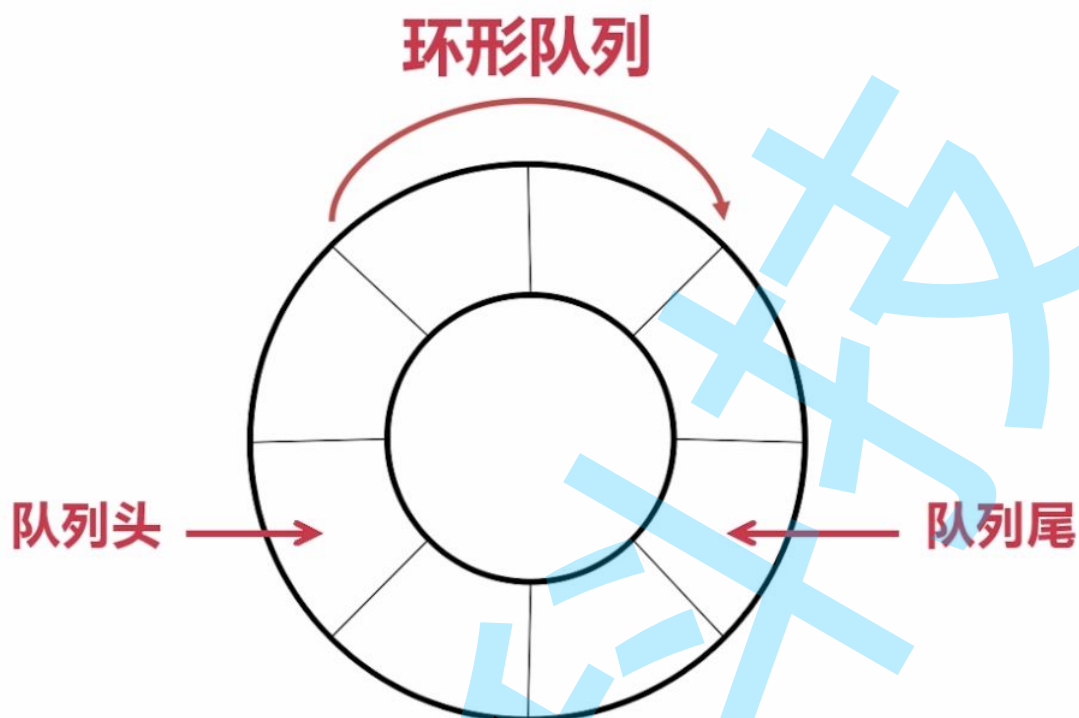
1  class Node {
2      int val;
3      Node next;
4
5      Node(int val, Node next) {
6          this.val = val;

```

```
7         this.next = next;
8     }
9
10    Node(int val) {
11        this(val, null);
12    }
13 }
14
15 public class MyQueue {
16     private Node head = null;
17     private Node tail = null;
18     private int size = 0;
19
20     public void offer(int v) {
21         Node node = new Node(v);
22         if (tail == null) {
23             head = node;
24         } else {
25             tail.next = node;
26         }
27         tail = node;
28         size++;
29     }
30
31     public int poll() {
32         if (size == 0) {
33             throw new RuntimeException("队列为空");
34         }
35
36         Node oldHead = head;
37         head = head.next;
38         if (head == null) {
39             tail = null;
40         }
41         size--;
42         return oldHead.val;
43     }
44
45     public int peek() {
46         if (size == 0) {
47             throw new RuntimeException("队列为空");
48         }
49
50         return head.val;
51     }
52
53     public boolean isEmpty() {
54         return size == 0;
55     }
56
57     public int size() {
58         return size;
59     }
60 }
```

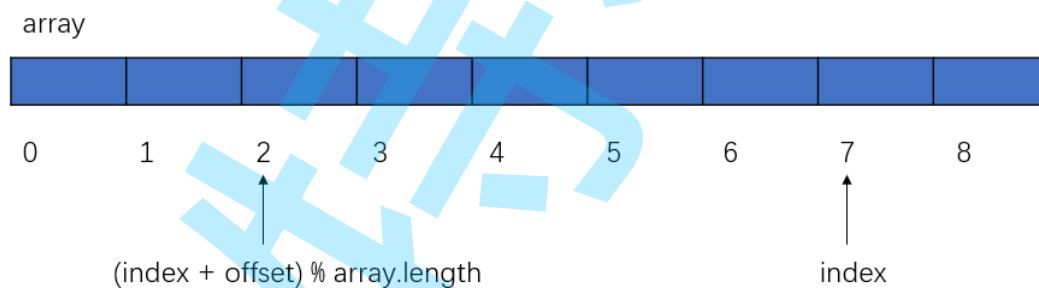
2.3 循环队列

实际中我们有时还会使用一种队列叫循环队列。如操作系统课程讲解生产者消费者模型时就会使用循环队列。环形队列通常使用数组实现。



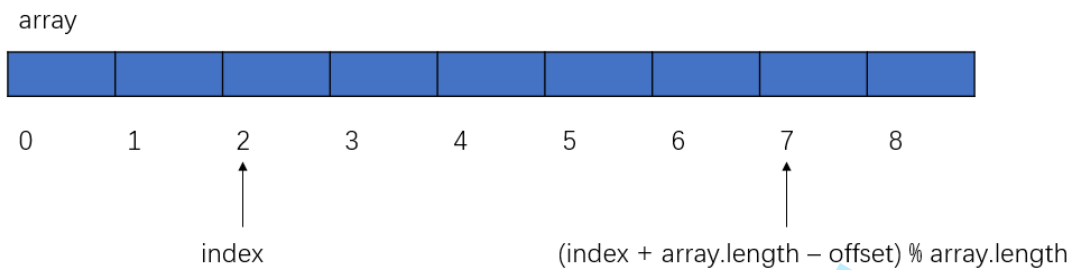
数组下标循环的小技巧

1. 下标最后再往后(offset 小于 array.length): $\text{index} = (\text{index} + \text{offset}) \% \text{array.length}$



index = 7
array.length = 9
offset = 4
结果 = 2

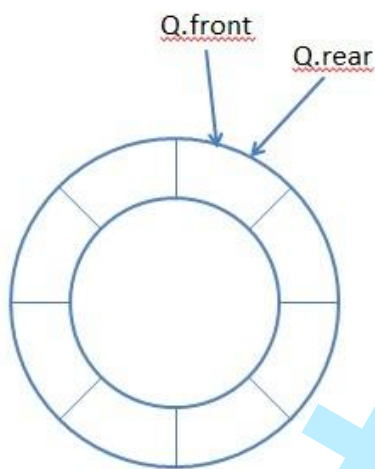
2. 下标最前再往前(offset 小于 array.length): $\text{index} = (\text{index} + \text{array.length} - \text{offset}) \% \text{array.length}$



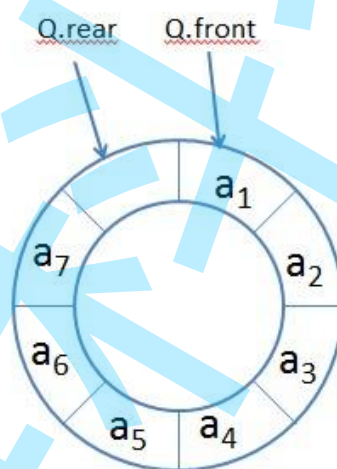
index = 2
array.length = 9
offset = 4
结果 = 7

如何区分空与满

1. 通过添加 size 属性记录
2. 保留一个位置



(a) 空的循环队列



(b) 满的循环队列

为了使用 $\text{Q.rear} = \text{Q.front}$ 来区别是队空还是队满，我们常常认为出现左图时的情况即为队满的情况，此时：
 $\text{rear} + 1 = \text{front}$

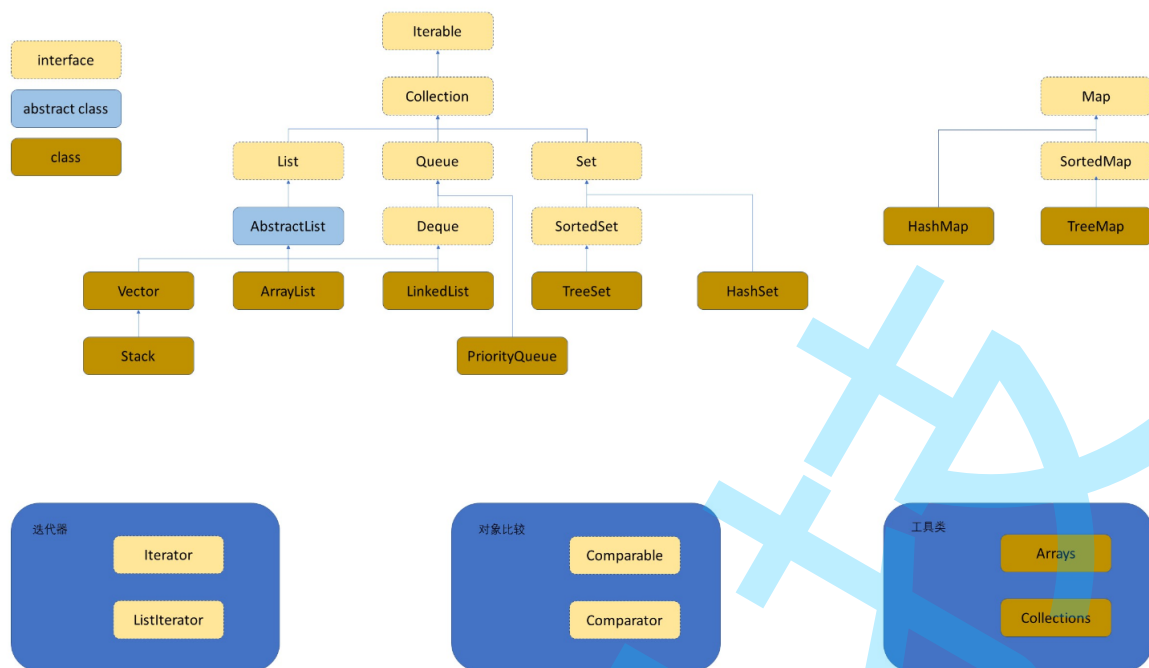
http://blog.csdn.net/zhang_xinxu

3. 双端队列 (Deque)

3.1 概念

双端队列 (deque) 是指允许两端都可以进行入队和出队操作的队列，deque 是 “double ended queue” 的简称。那就说明元素可以从队头出队和入队，也可以从队尾出队和入队。

4. java 中的栈和队列



Stack

方法	解释
E push (E item)	压栈
E pop ()	出栈
E peek ()	查看栈顶元素
boolean empty ()	判断栈是否为空

Queue

错误处理	抛出异常	返回特殊值
入队列	<code>add(e)</code>	<code>offer(e)</code>
出队列	<code>remove()</code>	<code>poll()</code>
队首元素	<code>element()</code>	<code>peek()</code>

Deque

头部/尾部	头部元素 (队首)		尾部元素 (队尾)	
错误处理	抛出异常	返回特殊值	抛出异常	返回特殊值
入队列	<code>addFirst(e)</code>	<code>offerFirst(e)</code>	<code>addLast(e)</code>	<code>offerLast(e)</code>
出队列	<code>removeFirst()</code>	<code>pollFirst()</code>	<code>removeLast()</code>	<code>pollLast()</code>
获取元素	<code>getFirst()</code>	<code>peekFirst()</code>	<code>getLast()</code>	<code>peekLast()</code>

5. 面试题

1. 括号匹配问题。 [OJ链接](#)

2. 用队列实现栈。 [OJ链接](#)
3. 用栈实现队列。 [OJ链接](#)
4. 实现一个最小栈。 [OJ链接](#)
5. 设计循环队列。 [OJ链接](#)

内容重点总结

- 掌握栈和队列的数据结构知识并做简单实现
- 掌握 java 中栈、队列、双端队列的使用
- 完成面试题

课后作业

- 博客总结数据结构知识
- 代码完成栈和队列的简易实现
- 通过面试题进行栈和队列的应用练习