数据结构：栈

作者：matthewhan
原文链接：https://ld246.com/article/1587453059662
来源网站：链滴

介绍

last-in-first-out，后进先出是它最大的特点。class Stack<E> extends Vector<E>作为Vector的子。

Vector底层使用数组存储数据，所以Stack也是如此。

Stack类**自身**的一些方法：

<figure class="md-table-fig" cid="n36" mdtype="table"><table class="md-table"><thead><tr class="md-end-block" cid="n37" mdtype="table_row"><th>方法</th><th>方法描述</th></tr></thead><tbody><tr class="md-end-block" cid="n40" mdtype="table_row"><td><code>boolean empty()</code></td><td>测试堆栈是否为空。</td></tr><tr class="md-end-block" cid="n43" mdtype="table_row"><td><code>Object peek()</code></td><td>查看堆栈顶部的对象，但不从堆栈中移除它。</td></tr><tr class="md-end-block" cid="n46" mdtype="table_row"><td><code>Object pop()</code

/span></td><td>移除堆栈顶部的对象，并作为此函数的值返回该对象。</td></tr><tr class="md-end-block" cid="n49" mdtype="table_row"><td><span c ass="td-span" cid="n50" mdtype="table_cell"> code>Object push(Object element)</code></td><td>把项压入堆栈 部。</td></tr><tr class="md-end-block" cid="n52" mdtype="table_row"> td><span md-inline="code" spellche k="false"><code>int search(Object element)</code></td><td>返回 象在堆栈中的位置，以 1 为基数。</td></tr></tbody></table></figure>

其中search方法比较特殊，返回的是该元素的位置，但是从1开始的，这个不是数组下标。源码里是样写的：

```
public synchronized int search(Object o) {
    int i = lastIndexOf(o);

    if (i >= 0) {
        return size() - i;
    }
    return -1;
}
```

如果该栈不存在任何元素，使用pop和peek方法会报NPE。

以上方法都是用了 synchronized进行修饰，确保是线程同步。

父类方法

留在Vector篇章记录。