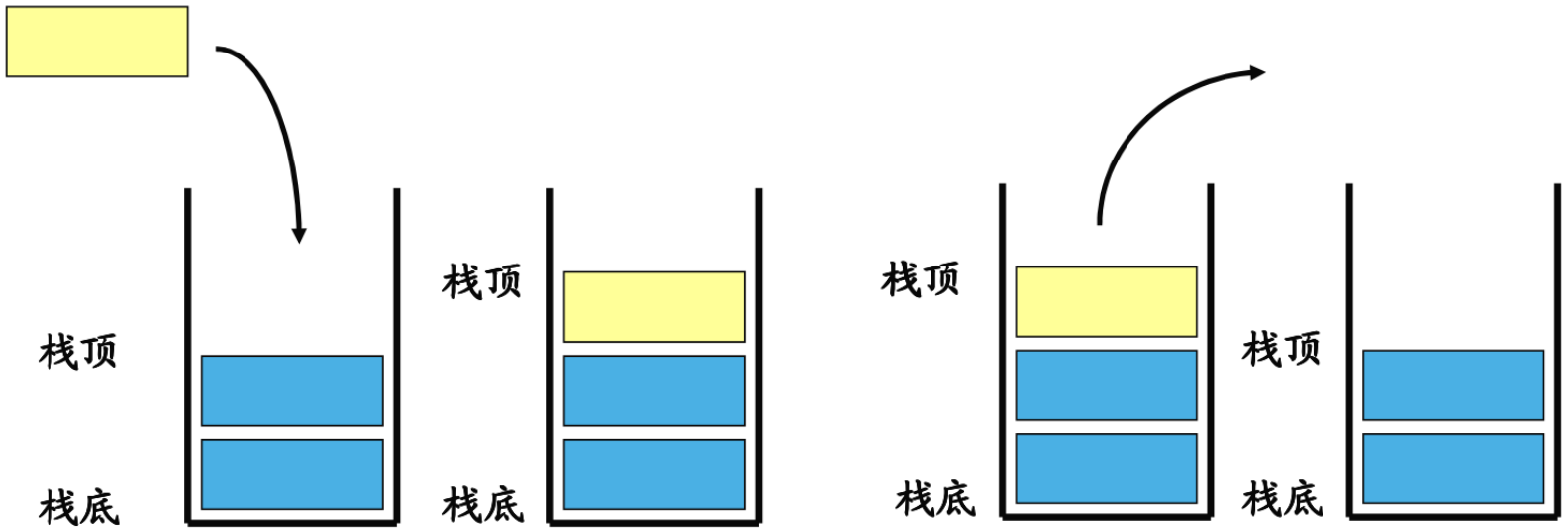
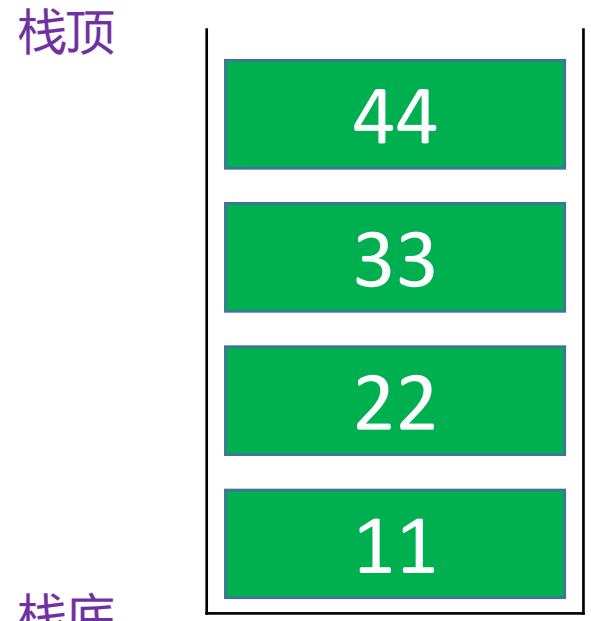
栈

栈（Stack）

- 栈是一种特殊的线性表，只能在一端进行操作
  - 往栈中添加元素的操作，一般叫做 push，入栈
  - 从栈中移除元素的操作，一般叫做 pop，出栈（只能移除栈顶元素，也叫做：弹出栈顶元素）
  - 后进先出的原则，Last In First Out，LIFO

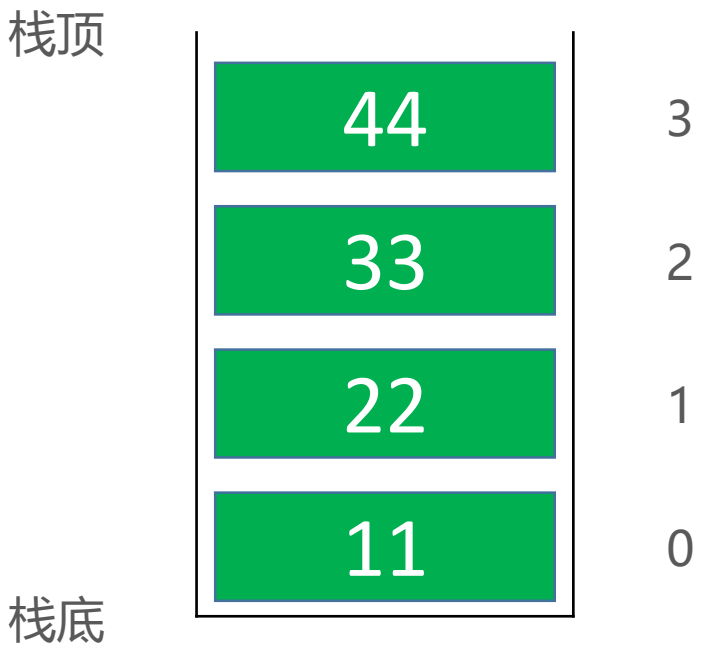




- 注意：这里说的“栈”与内存中的“栈空间”是两个不同的概念

栈的接口设计

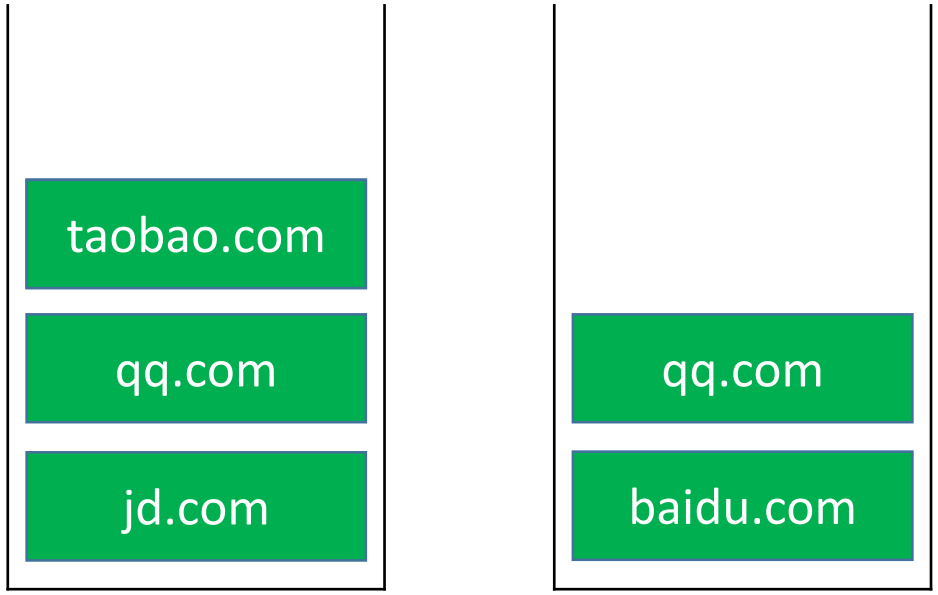
- `int size();  // 元素的数量`
- `boolean isEmpty();  // 是否为空`
- `void push(E element);  // 入栈`
- `E pop();  // 出栈`
- `E top();  // 获取栈顶元素`
- `void clear();  // 清空`



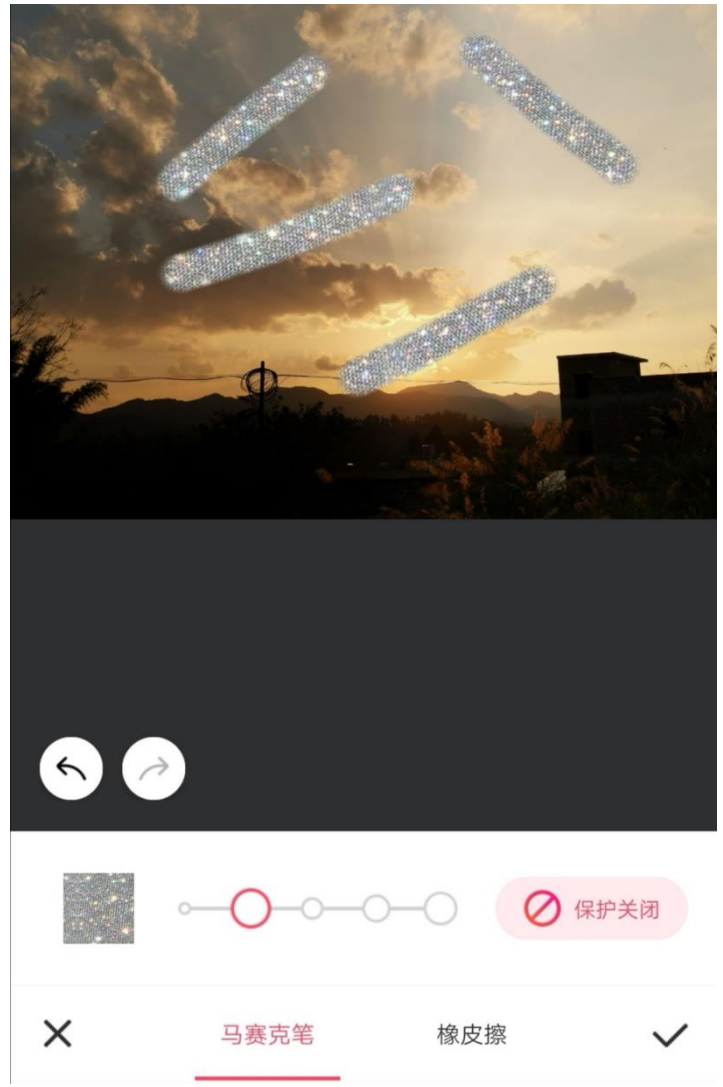
- 栈的内部实现是否可以直接利用以前学过的数据结构?
  - 动态数组、链表

栈的应用 – 浏览器的前进和后退

- 输入 jd.com
- 输入 qq.com
- 输入 baidu.com
- 后退
- 后退
- 前进
- 输入 taobao.com



- 类似的应用场景
  - 软件的撤销（Undo）、恢复（Redo）功能



练习 - 有效的括号

- https://leetcode-cn.com/problems/valid-parentheses/solution/

给定一个只包括 `'('`，`')'`，`'{'`，`'}'`，`'['`，`']'` 的字符串，判断字符串是否有效。

有效字符串需满足：

1. 左括号必须用相同类型的右括号闭合。
2. 左括号必须以正确的顺序闭合。

注意空字符串可被认为是有效字符串。

示例 1:

```
输入：  "()"
输出：  true
```

示例 2:

```
输入：  "()[]{}"
输出：  true
```

示例 3:

```
输入：  "(]"
输出：  false
```

示例 4:

```
输入：  "([)]"
输出：  false
```

示例 5:

```
输入：  "{[]}"
输出：  true
```

练习 - 有效的括号

1. 遇见左字符，将左字符入栈
2. 遇见右字符

- 如果栈是空的，说明括号无效
- 如果栈不为空，将栈顶字符出栈，与右字符之匹配

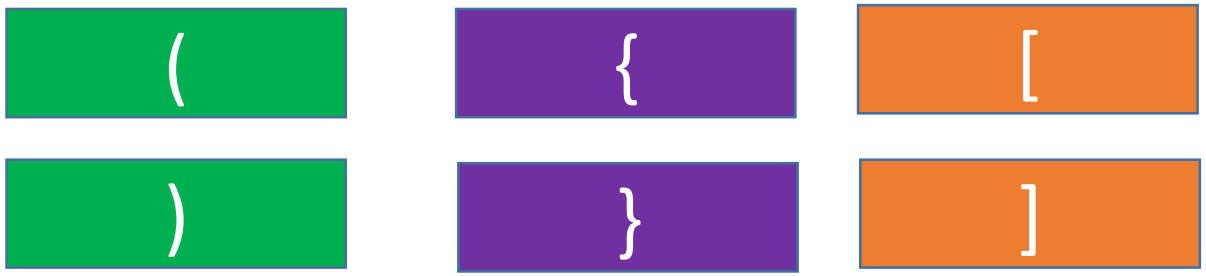
✓ 如果左右字符不匹配，说明括号无效

✓ 如果左右字符匹配，继续扫描下一个字符

3. 所有字符扫描完毕后

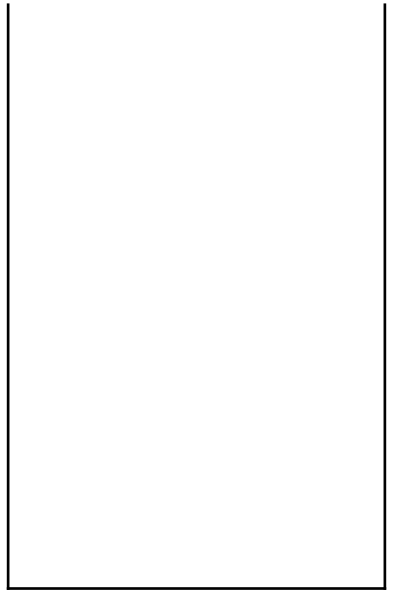
✓ 栈为空，说明括号有效

✓ 栈不为空，说明括号无效



■ 举例

- ( ) [{ }]
- {] }
- { } (



作业

- 括号的分数
  - https://leetcode-cn.com/problems/score-of-parentheses

- 逆波兰表达式求值
  - https://leetcode-cn.com/problems/evaluate-reverse-polish-notation/

- 基本计算器
  - https://leetcode-cn.com/problems/basic-calculator/comments/