# 020 Valid Parentheses

2018年3月29日 10:17

## Question:

Given a string containing just the characters '(', ')', '{', '}', '[' and ']', determine if the input string is valid.
The brackets must close in the correct order, "()" and "()[]{}" are all valid but "(]" and "([)]" are not.

来自 <https://leetcode.com/problems/valid-parentheses/description/>
给定一个只包括 '(', ')', '{', '}', '[', ']' 的字符串，判断字符串是否有效。
括号必须以正确的顺序关闭，"()" 和 "()[]{}" 是有效的但是 "(]" 和 "([)]" 不是。

## Solution for Python3:

```python
from collections import deque
class Solution:
    def isValid(self, s):
        """
        :type s: str
        :rtype: bool
        """
        d = {')':'(','}':'{',']':'['}
        deq = deque()
        for i in s:
            if i in d.values():
                deq.append(i)
            elif (len(deq) > 0) and d[i] == deq[-1]:
                deq.pop()
            else:
                return False
        return len(deq) == 0
```

## Solution for C++:

```cpp
class Solution {
public:
    bool isValid(string s) {
        stack<char> st;
        unordered_map<char, char> map;
        map['('] = ')';
        map['{'] = '}';
        map['['] = ']';
        for (int i = 0; i < s.length(); i++) {
            if (s[i] == '(' || s[i] == '{' || s[i] == '[') {
                st.push(s[i]);
            } else {
                if(!st.empty() && map[st.top()] == s[i]) {
                    st.pop();
                } else {
                    return false;
```

```
                    }
                }
            }
            return st.empty();
        }
    };
```

## Appendix:

**字典创建，collections模块中双端队列deque的使用**

**append(), pop(), appendleft(), popleft()，clear(), count(), insert(i,x), extend(iterable), extendleft(), reverse()翻转，rotate(n=1)旋转，len(deque)**