括号的有效性

```
class Solution {
public:
  bool isValid(std::string s) {
    std::stack<char> st;
    for (auto ch : s) {
      if (ch == '(' || ch == '{' || ch == '[') {
        st.push(ch);
      } else {
        if (st.size() && leftOf(ch) == st.top()) {
          st.pop();
        } else {
          return false;
        }
      }
    }

    return st.empty();
  }

private:
  char leftOf(char ch) {
    if (ch == ')') {
      return '(';
    } else if (ch == '}') {
      return '{';
    } else {
      return '[';
    }
  }
};
```

平衡括号串

给你输入一个字符串s，你可以在其中的任意位置插入左括号(或者右括号)，请问你最少需要几次插入才能使得s变成一个合法的括号串?

分析:

1. 只需统计左括号和右括号相差的个数即可。

```
class Solution {
public:
  int minAddToMakeValid(std::string s) {
    int res = 0, need = 0;
    for (auto ch : s) {
```

```
      if (ch == '(') {
        need++;  // 需要一个右括号
      }
      if (ch == ')') {
        need--;  // 减少一个有括号需求
        if (need == -1) {
          need = 0;
          res++;
        }
      }
    }
    return res + need;
  }
};
```

平衡括号串

> 现在假设 1 个左括号需要匹配 2 个右括号才叫做合法的括号组合，那么给你输入一个括号串s，请问你如何计算使得s合法的最小插入次数呢?

分析：

1. 当存在一个左括号时，需要两个右括号，如果对右括号的需求量为奇数，就需要插入一个右括号；
2. 当右括号超过左括号时，每两个右括号需要一个左括号，所以`need = 1`，保证当前需求不是从0开始。

```
class Solution {
public:
  int minInsertions(std::string s) {
    int need = 0;
    int res  = 0;

    for (auto ch : s) {
      if (ch == '(') {
        need += 2;
        if (need % 2 == 1) {
          res++;
          need--;
        }
      }

      if (ch == ')') {
        need--;
        if (need == -1) {  // 右括号太多了
          res++;
          need = 1;
        }
      }
    }
```

```
    return need + res;
  }
};
```