> 给你一个字符串 `s` ，每一次操作你都可以在字符串的任意位置插入任意字符。
> 请你返回让 `s` 成为回文串的 最少操作次数 。
> 「回文串」是正读和反读都相同的字符串。

分析： 定义dp数组： **dp[i][j]表示s[i..j]至少经过dp[i][j]次操作才能改变成回文串。** base case: 当`i == j`时， `dp[i][j] == 0`; 状态转移：

1. 当`s[i] == s[j]`时，`dp[i][j] = dp[i+1][j-1]`;
2. 当不相等时，`dp[i][j] = std::min(dp[i][j-1], dp[i+1][j])+1`

```
class Solution {
public:
  int minInsertions(std::string s) {
    int len = s.size();
    if (1 >= len) {
      return 0;
    }

    std::vector<std::vector<int>> dp =
        std::vector<std::vector<int>>(len + 1, std::vector<int>(len + 1,
0));

    // 计算dp数组
    for (int i = len - 2; i >= 0; i--) {
      for (int j = i + 1; j < len; j++) {
        if (s[i] == s[j]) {
          dp[i][j] = dp[i + 1][j - 1];
        } else {
          dp[i][j] = 1 + std::min(dp[i][j - 1], dp[i + 1][j]);
        }
      }
    }

    return dp[0][len-1];
  }
};
```