# 题目

给你一个字符串s ，找出其中最长的回文子序列，并返回该序列的长度。

子序列定义为：不改变剩余字符顺序的情况下，删除某些字符或者不删除任何字符形成的一个序列。

示例 1：

输入：s = "bbbab"

输出：4

解释：一个可能的最长回文子序列为 "bbbb" 。

示例 2：

输入：s = "cbbd"

输出：2

解释：一个可能的最长回文子序列为 "bb" 。

提示：

1 <= s.length <= 1000

s 仅由小写英文字母组成

# 分析

## 方法一：动态规划

**思路：**

**代码：**

class Solution {

public:

int longestPalindromeSubseq(string s) {

int n = s.length();

vector<vector<int>> dp(n, vector<int>(n));

for (int i = n - 1; i >= 0; i--) {

dp[i][i] = 1;

char c1 = s[i];

for (int j = i + 1; j < n; j++) {

char c2 = s[j];

if (c1 == c2) {

dp[i][j] = dp[i + 1][j - 1] + 2;

} else {

dp[i][j] = max(dp[i + 1][j], dp[i][j - 1]);

}

}

}

return dp[0][n - 1];

}

};

**复杂度分析：**

时间复杂度：O(n^2)，其中n是字符串s的长度。动态规划需要计算的状态数是O(n^2)。

空间复杂度：O(n^2)，其中n是字符串s的长度。需要创建二维数组dp，空间是O(n^2)。