LeetCode 第 70 号问题：爬楼梯

> *本文首发于公众号「图解面试算法」，是 图解 LeetCode 系列文章之一.*
>
> *同步博客：https://www.algomooc.com*

题目来源于 LeetCode 上第 70 号问题：爬楼梯。题目难度为 Easy。

题目描述

假设你正在爬楼梯。需要 `n` 阶你才能到达楼顶。

每次你可以爬 `1` 或 `2` 个台阶。你有多少种不同的方法可以爬到楼顶呢?

注意：给定 n 是一个正整数。

示例1

> *输入： 2*
>
> *解释： 有两种方法可以爬到楼顶。*
>
> 1. `1 阶 + 1 阶`
> 2. *2 阶*

题目解析

试着倒推想一下,就能发现这个问题可以被分解为一些包含最优子结构的子问题,它的最优解可以从其子问题 的最优解来有效地构建,因此我们可以使用 `动态规划` 解决这个问题.

第 `i` 阶可以由以下两种方法得到：

在第 `(i - 1)` 阶后向上爬 1 阶。

在第 `(i - 2)` 阶后向上爬 2 阶

所以到达第 `i` 阶的方法总数就是到第 `(i - 1)` 阶和第 `(i - 2)` 阶的方法数之和。

`dp[i]dp[i]` 表示能到达第 `i` 阶的方法总数,那么DP推导公式就是:

> *$$ dp[i] = dp[i - 1] + dp[i - 2] $$*

动画理解

 Animation

参考代码

```
/**
 * JavaScript 描述
 */
var climbStairs = function(n) {
```

```
    let temp = new Array(n+1);
    temp[1] = 1;
    temp[2] = 2;
    for (let i = 3; i < temp.length; i++) {
        temp[i] = temp[i-1] + temp[i-2];
    }
    return temp[n];
}
```

复杂度分析

- 时间复杂度：`O(n)`，单循环到 n。
- 空间复杂度：`O(n)`，dpdp 数组用了 n 的空间。

进一步优化

根据推导公式不难发现,我们要求的结果就是数组的最后一项,而最后一项又是前面数值叠加起来的,那么我们只需要两个变量保存 `i - 1` 和 `i - 2` 的值就可以了.

```
/**
 * JavaScript 描述
 */
var climbStairs = function(n) {
    if (n == 1) {
        return 1;
    }
    let first = 1,
        second = 2;
    for (let i = 3; i <= n; i++) {
       let third = first + second;
       first = second;
       second = third;
    }
    return second;
}
```

复杂度分析

- 时间复杂度：O(n)，单循环到 n。
- 空间复杂度：O(1)，用到了常量的空间。