

# Assignment #B: dp

Updated 1448 GMT+8 Nov 18, 2025

2025 fall, Compiled by 郭旭杰、化学与分子工程学院

账户: OpenJudge: 25n2500011906, 昵称: 郭旭杰

LeetCode/CodeForces/Luogu/sunnywhy: LittleBeetroot

## 说明:

- 1) 请把每个题目解题思路(可选), 源码Python, 或者C++ (已经在Codeforces/Openjudge上AC), 截图(包含Accepted), 填写到下面作业模版中(推荐使用 typora <https://typoraio.cn>, 或者用word)。AC 或者没有AC, 都请标上每个题目大致花费时间。
- 2) 提交时候先提交pdf文件, 再把md或者doc文件上传到右侧"作业评论"。Canvas需要有同学清晰头像、提交文件有pdf、"作业评论"区有上传的md或者doc附件。
- 3) 如果不能在截止前提交作业, 请写明原因。

## 1. 题目

### LuoguP1255 数楼梯

dp, bfs, math, brute force, <https://www.luogu.com.cn/problem/P1255>

耗时: 15min

思路: 直接不容易看出思路, 但是列举前几项, 发现都是斐波那契数列里面的对应项, 迭代转化为简单的斐波那契数列问题就可以秒杀了。

代码:

```
a = 0
b = 1
js = []
for _ in range(5000):
    a += b
    js.append(a)
    b += a
    js.append(b)

n = int(input())
print(js[n - 1])
```

代码运行截图 (至少包含有"Accepted")

应用 >> 题库 题单 比赛 记录 讨论 专栏

洛谷 / 评测记录 / 评测详情

R247989338 记录详情

编程语言 Python 3 代码长度 133B 用时 251ms 内存 8.82MB

测试点信息 源代码

测试点信息

#1 AC 25ms/8.57MB	#2 AC 25ms/8.79MB	#3 AC 26ms/8.63MB	#4 AC 25ms/8.80MB	#5 AC 25ms/8.57MB	#6 AC 24ms/8.79MB	#7 AC 26ms/8.82MB
#8 AC 24ms/8.66MB	#9 AC 26ms/8.63MB	#10 AC 25ms/8.79MB				

LittleBeetroot

所属题目 P1255 数楼梯

评测状态 Accepted

评测分数 100

提交时间 2025-11-18 15:51:10

应用 >> 题库 题单 比赛 记录 讨论 专栏

洛谷 / 评测记录 / 评测详情

R247989338 记录详情

编程语言 Python 3 代码长度 133B 用时 251ms 内存 8.82MB

测试点信息 源代码

源代码 [复制](#)

```
a = 0
b = 1
js = []
for _ in range(5000):
    a += b
    js.append(a)
    b += a
    js.append(b)

n = int(input())
print(js[n - 1])
```

LittleBeetroot

所属题目 P1255 数楼梯

评测状态 Accepted

评测分数 100

提交时间 2025-11-18 15:51:10

在洛谷，  
享受 Coding 的快乐

关于洛谷 | 帮助中心 | 用户协议 | 联系我们

小黑屋 | 图片放逐 | 社区规则 | 招贤纳士

Developed by the Luogu Dev Team

2013-2025, © 洛谷

增值电信业务经营许可证 沪B2-20200477

沪ICP备18008322号 All rights reserved.

27528: 跳台阶

dp, <http://cs101.openjudge.cn/practice/27528/>

耗时: 5min

思路：“狄贵”暗示使用递归；N<=25说明N较小，可以采取先使用迭代把所有解存储在列表中，输入N之后再直接从列表中取出答案。

代码：

```
js = [1]
for _ in range(25):
    js.append(sum(js) + 1)

n = int(input())
print(js[n - 1])
```

代码运行截图 (至少包含有"Accepted")

OpenJudge

题目ID, 标题, 描述

25n2500011906 信箱 账号

CS101 / 题库 (包括计概、数算题目)

题目

排名

状态

提问

27528:跳台阶

查看

提交

统计

提问

总时间限制: 1000ms 内存限制: 65536kB

描述

理工教学楼总共有N级台阶，狄贵同学每一步可以走的台阶数目可以是1、2、3、...、N-1、N中的任意一个。请问狄贵可以有多少种不同的走法走上这N级台阶。

输入

总共一行输入，输入台阶的阶数N。其中，1 <= N <= 25。

输出

多少种不同的走法走上N级台阶。

样例输入

3

样例输出

4

查看

提交

统计

提问

全局题号 27528

添加于 2024-10-23

提交次数 282

尝试人数 217

通过人数 217

你的提交记录

#	结果	时间
2	Accepted	2025-11-18
1	Accepted	2025-11-18

©2002-2022 POJ 京ICP备20010980号-1

English 帮助 关于

OpenJudge

题目ID, 标题, 描述

25n2500011906

信箱

账号

CS101 / 题库（包括计概、数算题目）

题目

排名

状态

提问

#50891450提交状态

查看

提交

统计

提问

状态: Accepted

源代码

```
js = [1]
for _ in range(25):
    js.append(sum(js) + 1)

n = int(input())
print(js[n - 1])
```

基本信息

#: 50891450

题目: 27528

提交人: 25n2500011906(Little(25n2500011906))

内存: 3608kB

时间: 21ms

语言: Python3

提交时间: 2025-11-18 15:55:00

©2002-2022 POJ 京ICP备20010980号-1

English

帮助

关于

## M23421: 《算法图解》小偷背包问题

dp, <http://cs101.openjudge.cn/pctbook/M23421/>

耗时: 1h

思路: 这题是动态规划 (dp) 的鼻祖, 可以结合dfs完成。

模拟了半天没成功, 随手改了一个js变成js0提交就AC了。真是无语了。

代码:

```
n, b = list(map(int, input().split()))
prices = list(map(int, input().split()))
weights = list(map(int, input().split()))
js = []
for i in range(n):
    js.append([prices[i], weights[i], False])
earns = [0]

def dfs(js0, b0):
    earn = 0
    if b0 >= js0[0][1]:
        js0[0][2] = True
        if len(js0) >= 2:
            dfs(js0[1:], b0 - js0[0][1])
        for j in js:
            if j[2] is True:
                earn += j[0]
        earns.append(earn)
    earn -= earn
```

```
js0[0][2] = False
if len(js0) >= 2:
    dfs(js0[1:], b0)

dfs(js, b)
print(max(earns))
```

代码运行截图 (至少包含有"Accepted")

OpenJudge

题目ID, 标题, 描述

25n2500011906

信箱

账号

CS101 / 计算思维算法实践

题目 排名 状态 提问

M23421: 《算法图解》小偷背包问题

查看 提交 统计 提问

总时间限制: 1000ms    内存限制: 65536kB

描述

这是《算法图解》[1]书中第9章动态规划的例子：一个小贼正在一家店里偷商品。

假设一种情况如下：

一个小偷背着一个可装4磅东西的背包。商场有三件物品分别为：  
价值3000美元重4磅的音响，价值2000美元重3磅的笔记本，价值1500美元重1磅的吉他。

问小偷应该怎样选择商品，才能使得偷取的价值最高？

[1]Grokking Algorithms by Aditya Bhargava, published by Manning Publications. Copyright © 2016 by Manning Publications.  
Simplified Chinese-language edition copyright © 2017 by Posts & Telecom Press.

全局题号 23421

添加于 2025-03-13

提交次数 133

尝试人数 99

通过人数 98

Other language versions

English

你的提交记录

#	结果	时间
1	Accepted	2025-11-20

## #50917926提交状态

[查看](#) [提交](#) [统计](#) [提问](#)

状态: **Accepted**

### 源代码

```
n, b = list(map(int, input().split()))
prices = list(map(int, input().split()))
weights = list(map(int, input().split()))
js = []
for i in range(n):
    js.append([prices[i], weights[i], False])
earns = [0]

def dfs(js0, b0):
    earn = 0
    if b0 >= js0[0][1]:
        js0[0][2] = True
        if len(js0) >= 2:
            dfs(js0[1:], b0 - js0[0][1])
        for j in js:
            if j[2] is True:
                earn += j[0]
        earns.append(earn)
        earn -= earn

    js0[0][2] = False
    if len(js0) >= 2:
        dfs(js0[1:], b0)

dfs(js, b)
print(max(earns))
```

### 基本信息

#: 50917926  
题目: M23421  
提交人: 25n2500011906(Little(25n2500011906))  
内存: 3620kB  
时间: 23ms  
语言: Python3  
提交时间: 2025-11-20 14:41:08

## M5.最长回文子串

dp, two pointers, string, <https://leetcode.cn/problems/longest-palindromic-substring/>

耗时: 1h

思路: 这是一个找回文序列的问题, 涉及双指针, 时间复杂度为 $O(n\log n)$ 。

应该直接对字符串进行切片, 注意如果挨个存储回文序列会导致超时, 因此需要记录目前最长的回文序列的长度 $lm$ , 这样当序列长度显然小于 $lm$ 时直接跳过, 节省算力。不断更新记录回文序列的列表 $res$ , 使得列表中存储的回文序列越来越长, 最后一个就是最长的回文序列。

不进行简化的算法时间复杂度为 $O(n^2)$ , 加之以数据较大, 会超时。

注意到数据较为庞大, 不能使用二维数组, 更不应该将字符串转化为列表, 否则会超时。

代码:

```
class Solution:
    def longestPalindrome(self, s: str) -> str:
```

```

n = len(s)
res = []
lm = 0
for i in range(n):
    for j in range(i, n + 1):
        if j - i >= lm:
            if s[i: j] == s[i: j][::-1]:
                res.append(s[i: j])
                lm = j - i
return res[-1]

```

代码运行截图 (至少包含有"Accepted")

The screenshot displays a submission page for a problem. The top section shows the submission status as "通过" (Accepted) with 142/142 test cases passed, a runtime of 2 hrs 30 m 23 s, and a submission time of 2025.11.23 21:33. Below this, there is a section for execution statistics, including a bar chart showing the distribution of execution times, with a peak around 30ms. The statistics also show a total runtime of 4801 ms, a hit rate of 5.24%, and a memory usage of 18.24 MB with a hit rate of 41.93%. The code section shows the Python3 code for the solution, which is a class Solution with a method longestPalindrome. The test results section shows two test cases, both of which are passed, with the input "babad" and the output "aba".

通过 142 / 142 个通过的测试用例 用时: 2 hrs 30 m 23 s  
LittleBeetroot 提交于 2025.11.23 21:33

面向在校学生的专享特惠  
完成认证享 7 折 Plus 会员, 享受更多学业及职业成长帮助

执行用时分布  
4801 ms | 击败 5.24%  
复杂度分析

消耗内存分布  
18.24 MB | 击败 41.93%

代码 | Python3

```

class Solution:
    def longestPalindrome(self, s: str) -> str:
        n = len(s)
        res = []
        lm = 0
        for i in range(n):
            for j in range(i, n + 1):
                if j - i >= lm:
                    if s[i: j] == s[i: j][::-1]:
                        res.append(s[i: j])
                        lm = j - i
        return res[-1]

```

测试用例 测试结果

通过 执行用时: 0 ms

Case 1 Case 2

输入  
s =  
"babad"

输出  
"aba"

预期结果  
"bab"

题库

提交记录

所有状态	所有语言	执行用时	消耗内存	备注
12 通过 2025.11.23	Python3	5217 ms	18.2 MB	
11 通过 2025.11.23	Python3	4801 ms	18.2 MB	[ 用时: 2 hrs 30 ... ]
10 超出时间限制 2025.11.23	Python3	N/A	N/A	
9 解答错误 2025.11.23	Python3	N/A	N/A	
8 执行出错 2025.11.23	Python3	N/A	N/A	
7 执行出错 2025.11.23	Python3	N/A	N/A	
6 超出时间限制 2025.11.23	Python3	N/A	N/A	
5 超出时间限制 2025.11.23	Python3	N/A	N/A	
4 超出时间限制 2025.11.20	Python3	N/A	N/A	

代码

Python3 智能模式

```
1 class Solution:
2     def longestPalindrome(self, s: str) -> str:
3
4         n = len(s)
5         res = []
6         lm = 0
7         for i in range(n):
8             for j in range(i, n + 1):
9                 if j - i >= lm:
10                    if s[i: j] == s[i: j][::-1]:
11                        res.append(s[i: j])
12                        lm = j - i
13         return res[-1]
14
```

## 474D. Flowers

dp, recursion 1700 <https://codeforces.com/problemset/problem/474/D>

耗时: 4h

思路: 可怕的算法, 我用组合数做了两个半小时, 一直TLE。无奈之下求助AI, 给出的算法我一遍一遍地打磨, 一个半小时后方恍然大悟。

递归时分为两种情况: 最后一个吃红花和最后一个吃白花。如果最后一个吃红花, 那么前面 $i - 1$ 朵花就有 $f[i - 1]$ 种吃法; 如果最后一个吃白花, 那么必然最后 $k$ 个吃的都是白花, 前面 $i - k$ 朵花共有 $f[i - k]$ 种吃法; 特别的, 如果 $i < k$ , 那么最后一个不可能吃白花。

注意计算得 $f[i]$ 之后及时mod, 有利于节省算力并输出正确结果。

采用预处理, 避免代码在执行中反复执行, 大大节约了时间。

sgm为前缀和, res的计算采取了前缀和作差的算法。

代码:

```
MOD = 10 ** 9 + 7

t, k = list(map(int, input().split()))

f = [0] * (10 ** 5 + 1)
f[0] = 1

for i in range(1, 10 ** 5 + 1):
    f[i] = f[i - 1]
    if i >= k:
        f[i] += f[i - k]
    # 当i>=k时, 即f[i] = f[i - 1] + f[i - k]
    f[i] %= MOD
```



```

sgm = [0] * (10 ** 5 + 1)
sgm[0] = f[0]
for i in range(1, 10 ** 5 + 1):
    sgm[i] = (sgm[i - 1] + f[i]) % MOD

for _ in range(t):
    a, b = list(map(int, input().split()))
    res = (sgm[b] - sgm[a - 1]) % MOD
    print(res)

```

代码运行截图 (至少包含有"Accepted")



[LittleBeetroot](#) | [Logout](#)  
 You have **+2761 Wow!**

[HOME](#)
[TOP](#)
[CATALOG](#)
[CONTESTS](#)
[GYM](#)
[PROBLEMSET](#)
[GROUPS](#)
[RATING](#)
[EDU](#)
[API](#)
[CALENDAR](#)
[HELP](#)



[PROBLEMS](#)
[SUBMIT CODE](#)
[MY SUBMISSIONS](#)
[STATUS](#)
[HACKS](#)
[ROOM](#)
[STANDINGS](#)
[CUSTOM INVOCATION](#)

General										
#	Author	Problem	Lang	Verdict	Time	Memory	Sent	Judged		
350534701	Practice: LittleBeetroot	474D - 15	PyPy 3-64	Accepted	484 ms	7848 KB	2025-11-24 22:54:18	2025-11-24 22:54:18	★	<a href="#">Compare</a>

→ [Source](#)

[Copy](#)

```

MOD = 10 ** 9 + 7

t, k = list(map(int, input().split()))

f = [0] * (10 ** 5 + 1)
f[0] = 1

for i in range(1, 10 ** 5 + 1):
    f[i] = f[i - 1]
    if i >= k:
        f[i] += f[i - k]
    f[i] %= MOD

sgm = [0] * (10 ** 5 + 1)
sgm[0] = f[0]
for i in range(1, 10 ** 5 + 1):
    sgm[i] = (sgm[i - 1] + f[i]) % MOD

for _ in range(t):
    a, b = list(map(int, input().split()))
    res = (sgm[b] - sgm[a - 1]) % MOD
    print(res)

```

[Click](#) to see test details

My Submissions							
#	When	Who	Problem	Lang	Verdict	Time	Memory
<a href="#">350534701</a>	Nov/25/2025 03:54 <sup>UTC+8</sup>	LittleBeetroot	<a href="#">D - Flowers</a>	PyPy 3-64	Accepted	484 ms	7800 KB
<a href="#">350486725</a>	Nov/24/2025 21:16 <sup>UTC+8</sup>	LittleBeetroot	<a href="#">D - Flowers</a>	PyPy 3-64	Accepted	171 ms	9700 KB
<a href="#">350486476</a>	Nov/24/2025 21:14 <sup>UTC+8</sup>	LittleBeetroot	<a href="#">D - Flowers</a>	PyPy 3-64	Runtime error on test 1	109 ms	2700 KB
<a href="#">350484957</a>	Nov/24/2025 21:03 <sup>UTC+8</sup>	LittleBeetroot	<a href="#">D - Flowers</a>	PyPy 3-64	Accepted	171 ms	20200 KB
<a href="#">350484647</a>	Nov/24/2025 21:00 <sup>UTC+8</sup>	LittleBeetroot	<a href="#">D - Flowers</a>	PyPy 3-64	Time limit exceeded on test 5	1500 ms	4500 KB
<a href="#">350483821</a>	Nov/24/2025 20:54 <sup>UTC+8</sup>	LittleBeetroot	<a href="#">D - Flowers</a>	PyPy 3-64	Time limit exceeded on test 5	1500 ms	18800 KB
<a href="#">350481454</a>	Nov/24/2025 20:38 <sup>UTC+8</sup>	LittleBeetroot	<a href="#">D - Flowers</a>	PyPy 3-64	Time limit exceeded on test 1	1500 ms	18400 KB
<a href="#">350480583</a>	Nov/24/2025 20:32 <sup>UTC+8</sup>	LittleBeetroot	<a href="#">D - Flowers</a>	PyPy 3-64	Time limit exceeded on test 5	1500 ms	18600 KB
<a href="#">350479793</a>	Nov/24/2025 20:26 <sup>UTC+8</sup>	LittleBeetroot	<a href="#">D - Flowers</a>	PyPy 3-64	Wrong answer on test 1	109 ms	17200 KB
<a href="#">350479691</a>	Nov/24/2025 20:25 <sup>UTC+8</sup>	LittleBeetroot	<a href="#">D - Flowers</a>	PyPy 3-64	Wrong answer on test 1	108 ms	17200 KB
<a href="#">350479624</a>	Nov/24/2025 20:25 <sup>UTC+8</sup>	LittleBeetroot	<a href="#">D - Flowers</a>	PyPy 3-64	Wrong answer on test 1	108 ms	17200 KB
<a href="#">350478671</a>	Nov/24/2025 20:18 <sup>UTC+8</sup>	LittleBeetroot	<a href="#">D - Flowers</a>	PyPy 3-64	Wrong answer on test 2	139 ms	17900 KB
<a href="#">350478552</a>	Nov/24/2025 20:17 <sup>UTC+8</sup>	LittleBeetroot	<a href="#">D - Flowers</a>	PyPy 3-64	Wrong answer on test 1	61 ms	17200 KB
<a href="#">350478434</a>	Nov/24/2025 20:16 <sup>UTC+8</sup>	LittleBeetroot	<a href="#">D - Flowers</a>	PyPy 3-64	Wrong answer on test 1	93 ms	17200 KB
<a href="#">350478257</a>	Nov/24/2025 20:15 <sup>UTC+8</sup>	LittleBeetroot	<a href="#">D - Flowers</a>	PyPy 3-64	Wrong answer on test 1	77 ms	17200 KB

## M198.打家劫舍

dp, dfs, recursion <https://leetcode.cn/problems/house-robber/>

耗时：3h

思路：我一开始想到openjudge上“马走日”的问题，想到数据总量很小，可以先把所有偷到不能再偷的可能——遍历存到列表里面，然后取最大值。感觉可行但是实际操作一直WA。

后来查看题解学习了dp解法，即对于每一间房屋，小偷有“偷”和“不偷”两种选择，且如果偷这间房屋就不能偷前一间房屋(dp[i - 2] + nums[i])，如果不偷就考虑一直偷到前一间房屋所能得到的最大金额(dp[i - 1])。两种方法相比较取其优，一直这样下去直到偷得不能再偷为止。

刚才又通过班级群里交流、询问AI等方式学习到了可行易懂的dfs算法，就是遍历所有不会触发警报的可能，存储在列表中，再取列表中元素的最大值即可。但是实际操作结果会MLE。（时间复杂度也已经达到了可怕的2\*\*n了）

代码：

dp

```

class Solution:
    def rob(self, nums: List[int]) -> int:
        n = len(nums)
        if n == 1:
            return nums[-1]
        exit
        dp = [0 for _ in range(n)]
        dp[0] = nums[0]
        dp[1] = max(nums[0], nums[1])
        for i in range(2, n):
            dp[i] = max(dp[i - 2] + nums[i], dp[i - 1])
        return dp[-1]

```

dfs

```

class solution:
    def rob(self, nums: List[int]) -> int:
        res = [0]

        def dfs(index, current_sum):
            # 终止条件: 索引超出房屋范围时, 将当前金额加入结果
            if index >= len(nums):
                res.append(current_sum)
                return
            # 选择偷窃当前房屋, 跳过下一间房屋
            dfs(index + 2, current_sum + nums[index])
            # 选择不偷窃当前房屋, 继续考虑下一间房屋
            dfs(index + 1, current_sum)

        # 从第0间房屋开始递归, 初始金额为0
        dfs(0, 0)
        res.sort(reverse=True)
        return res[0]

```

代码运行截图 (至少包含有"Accepted")

dp

题库

提交

02:17:06

Plus 会员

题目描述

通过

题解

提交记录

全部提交记录

通过

70 / 70 个通过的测试用例 用时: 2 hrs 17 m 6 s

LittleBeetroot 提交于 2025.11.25 03:01

官方题解

写题解

面向在校学生的专享特惠

完成认证享 7 折 Plus 会员, 享受更多学业及职业成长帮助

执行用时分布

0 ms | 击败 100.00%

消耗内存分布

17.44 MB | 击败 62.46%

复杂度分析

100%

50%

0%

1ms

2ms

3ms

4ms

代码 | Python3

```
class Solution:
    def rob(self, nums: List[int]) -> int:
        n = len(nums)
        if n == 1:
            return nums[-1]
        exit
        dp = [0 for _ in range(n)]
        dp[0] = nums[0]
```

查看更多

</> 代码

Python3

智能模式

```
1 class Solution:
2     def rob(self, nums: List[int]) -> int:
3         n = len(nums)
4         if n == 1:
5             return nums[-1]
6         exit
7         dp = [0 for _ in range(n)]
8         dp[0] = nums[0]
9         dp[1] = max(nums[0], nums[1])
10        for i in range(2, n):
11            dp[i] = max(dp[i - 2] + nums[i], dp[i - 1])
12        return dp[-1]
13
```

已存储

行 1, 列 1

测试用例

测试结果

通过

执行用时: 0 ms

Case 1

Case 2

输入

nums =

[1,2,3,1]

输出

4

预期结果

4

贡献测试用例

题库

提交

02:17:06

Plus 会员

题目描述

题解

提交记录

所有状态

所有语言

执行用时

消耗内存

备注

通过

1 分钟前

Python3

0 ms

17.4 MB

[用时: 2 hrs 17 m 6 s]

通过

9 小时前

Python3

0 ms

17.5 MB

通过

9 小时前

Python3

0 ms

17.5 MB

[用时: 20 hrs 21 m 54 s]

通过

9 小时前

Python3

2 ms

17.4 MB

[用时: 20 hrs 21 m 51 s]

解答错误

2025.11.23

Python3

N/A

N/A

解答错误

2025.11.20

Python3

N/A

N/A

</> 代码

Python3

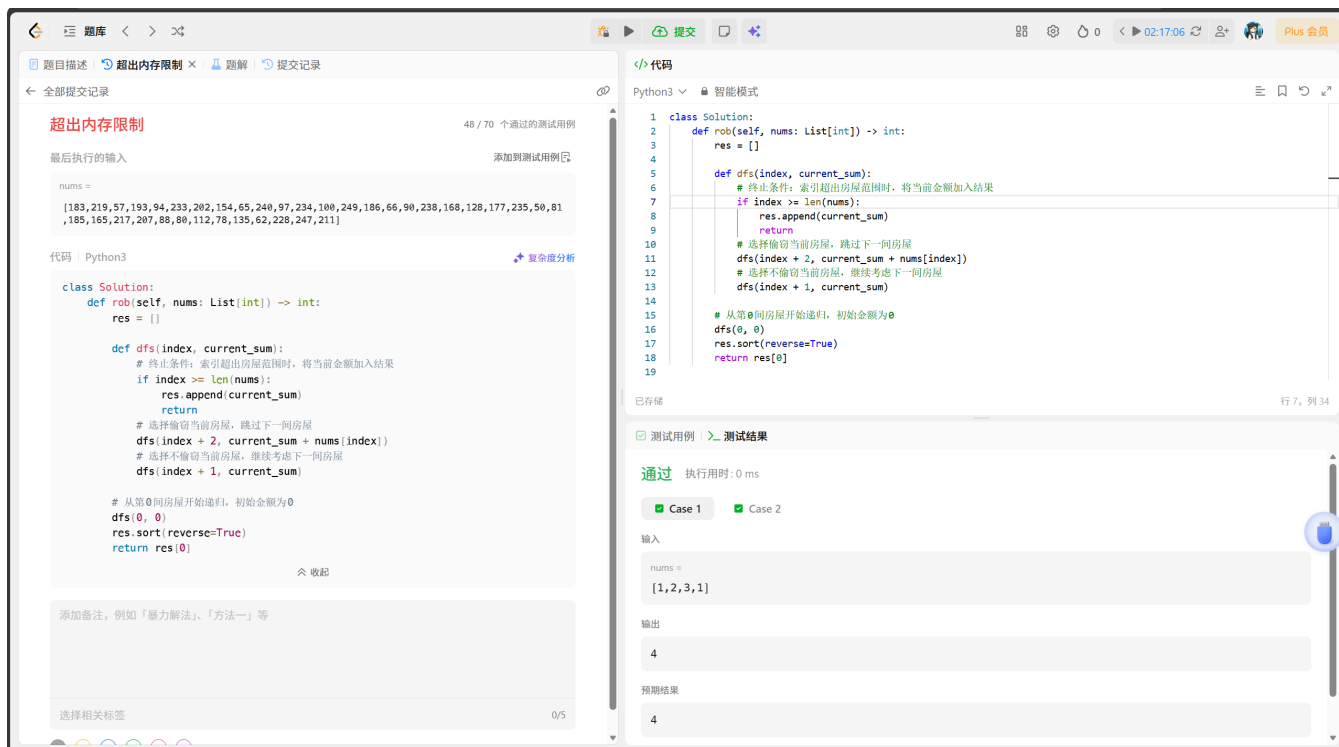
智能模式

```
1 class Solution:
2     def rob(self, nums: List[int]) -> int:
3         n = len(nums)
4         if n == 1:
5             return nums[-1]
6         exit
7         dp = [0 for _ in range(n)]
8         dp[0] = nums[0]
9         dp[1] = max(nums[0], nums[1])
10        for i in range(2, n):
11            dp[i] = max(dp[i - 2] + nums[i], dp[i - 1])
12        return dp[-1]
13
```

已存储

行 1, 列 1

dfs



## 时间复杂度

$$O(2^N)$$



## 2. 学习总结和收获

学习了dp, 感觉dp和dfs有很大区别, dfs难在程序的正确表达, 思路简单清晰; dp结合recursion恰恰相反, 程序好写, 正确思路难找。很多情况下, 有些题既可以用dfs又可以用dp, 如果找不到dp的思路就用dfs, 反之如果dfs代码过于难写或dfs明显超时就用dp, 有时还需要借助greedy算法。

如果作业题目简单, 有否额外练习题目, 比如: OJ“计概2024fall每日选做”、CF、LeetCode、洛谷等网站题目。

本周任务较为繁重, 只额外刷了两个题, 一个dp, 一个greedy

## T04117:简单的整数划分问题

math, dp, <http://cs101.openjudge.cn/pctbook/T04117/>

代码:

```
def div(n0, k0):
```

```
if k0 == 1:
    return 1
elif k0 == 2:
    return n0 // 2
else:
    res = 0
    i = 0
    while n0 - i * k0 - 1 >= 2:
        res += div(n0 - i * k0 - 1, k0 - 1)
        i += 1
    return res

def solve():
    n = int(input())
    sgm = 0
    for i in range(1, n + 1):
        sgm += div(n, i)
    print(sgm)

while True:
    try:
        if __name__ == '__main__':
            solve()
    except:
        break
```

代码运行截图：

T04117:简单的整数划分问题

查看

提交

统计

提问

总时间限制: 100ms    内存限制: 65536kB

描述

将正整数 $n$  表示成一系列正整数之和,  $n=n_1+n_2+...+n_k$ , 其中 $n_1 \geq n_2 \geq ... \geq n_k \geq 1$  ,  $k \geq 1$  。正整数 $n$  的这种表示称为正整数 $n$  的划分。正整数 $n$  的不同的划分个数称为正整数 $n$  的划分数。

输入

标准的输入包含若干组测试数据。每组测试数据是一个整数 $N(0 < N \leq 50)$ 。

输出

对于每组测试数据，输出 $N$ 的划分数。

样例输入

5

样例输出

7

提示

5, 4+1, 3+2, 3+1+1, 2+2+1, 2+1+1+1, 1+1+1+1+1

查看

提交

统计

提问

全局题号 **7215**

添加于 **2025-03-13**

提交次数 **241**

尝试人数 **57**

通过人数 **54**

你的提交记录

#	结果	时间
4	Accepted	2025-11-25
3	Wrong Answer	2025-11-25
2	Wrong Answer	2025-11-24
1	Runtime Error	2025-11-24

## #50985869提交状态

[查看](#) [提交](#) [统计](#) [提问](#)

状态: **Accepted**

源代码

```
def div(n0, k0):
    if k0 == 1:
        return 1
    elif k0 == 2:
        return n0 // 2
    else:
        res = 0
        i = 0
        while n0 - i * k0 - 1 >= 2:
            res += div(n0 - i * k0 - 1, k0 - 1)
            i += 1
        return res

def solve():
    n = int(input())
    sgm = 0
    for i in range(1, n + 1):
        sgm += div(n, i)
    print(sgm)

while True:
    try:
        if __name__ == '__main__':
            solve()
    except:
        break
```

基本信息

#: 50985869  
题目: T04117  
提交人: 25n2500011906(Little(25n2500011906))  
内存: 3548kB  
时间: 274ms  
语言: Python3

提交时间: 2025-11-25 02:40:52

## 32C:Flea

math, greedy, 1700 <https://codeforces.com/problemset/problem/32/C>

代码:

```
def tpp(a0, b0):
    if a0 % b0 == 0:
        return a0
    else:
        return (a0 % b0) * (a0 // b0 + 1)

n, m, s = list(map(int, input().split()))
print(tpp(m, s) * tpp(n, s))
```

代码运行截图:



PROBLEMS SUBMIT CODE MY SUBMISSIONS STATUS HACKS ROOM STANDINGS CUSTOM INVOCATION

General									
#	Author	Problem	Lang	Verdict	Time	Memory	Sent	Judged	
350539030	Practice: LittleBeetroot	32C - 12	PyPy 3-64	Accepted	186 ms	0 KB	2025-11-24 23:54:37	2025-11-24 23:54:37	<div><div></div><div>Compare</div></div>

→ Source

Copy

```
def tpp(a0, b0):  
    if a0 % b0 == 0:  
        return a0  
    else:  
        return (a0 % b0) * (a0 // b0 + 1)  
  
n, m, s = list(map(int, input().split()))  
print(tpp(m, s) * tpp(n, s))
```

PROBLEMS SUBMIT CODE MY SUBMISSIONS STATUS HACKS ROOM STANDINGS CUSTOM INVOCATION

My Submissions							
#	When	Who	Problem	Lang	Verdict	Time	Memory
<a href="#">350539030</a>	Nov/25/2025 04:54 <sup>UTC+8</sup>	LittleBeetroot	<a href="#">C - Flea</a>	PyPy 3-64	Accepted	186 ms	0 KB
<a href="#">350538930</a>	Nov/25/2025 04:53 <sup>UTC+8</sup>	LittleBeetroot	<a href="#">C - Flea</a>	PyPy 3-64	Wrong answer on test 6	154 ms	0 KB
<a href="#">350538764</a>	Nov/25/2025 04:50 <sup>UTC+8</sup>	LittleBeetroot	<a href="#">C - Flea</a>	PyPy 3-64	Wrong answer on test 6	154 ms	0 KB
<a href="#">350466671</a>	Nov/24/2025 18:31 <sup>UTC+8</sup>	LittleBeetroot	<a href="#">C - Flea</a>	PyPy 3-64	Wrong answer on test 5	156 ms	0 KB