WEEK8

段雷

目　录

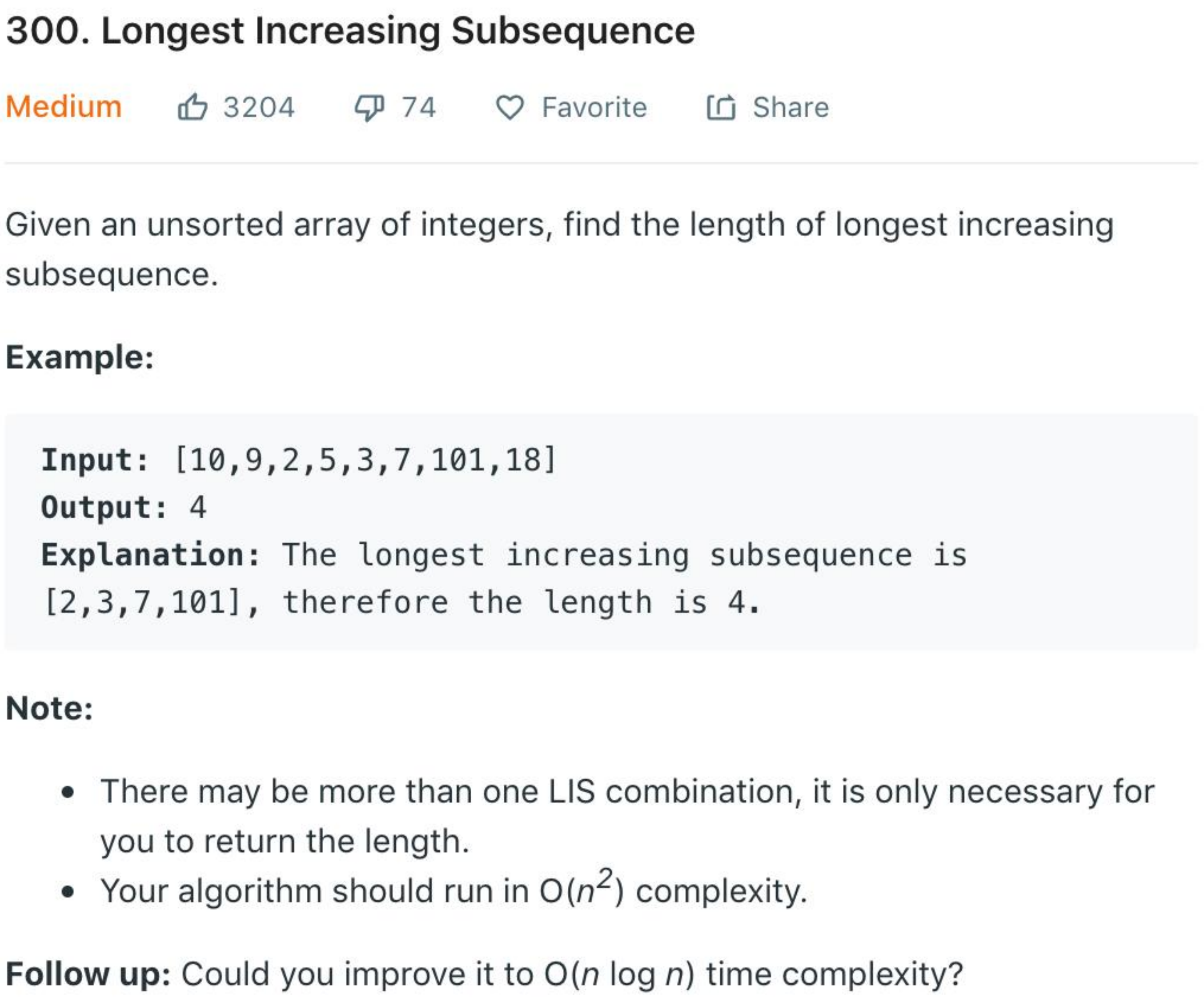
CONTENTS

01 题目讲解

02 背包问题

01题目讲解

Leetcode300



300. Longest Increasing Subsequence

Medium 👍 3204 👎 74 ♡ Favorite Share

Given an unsorted array of integers, find the length of longest increasing subsequence.

Example:

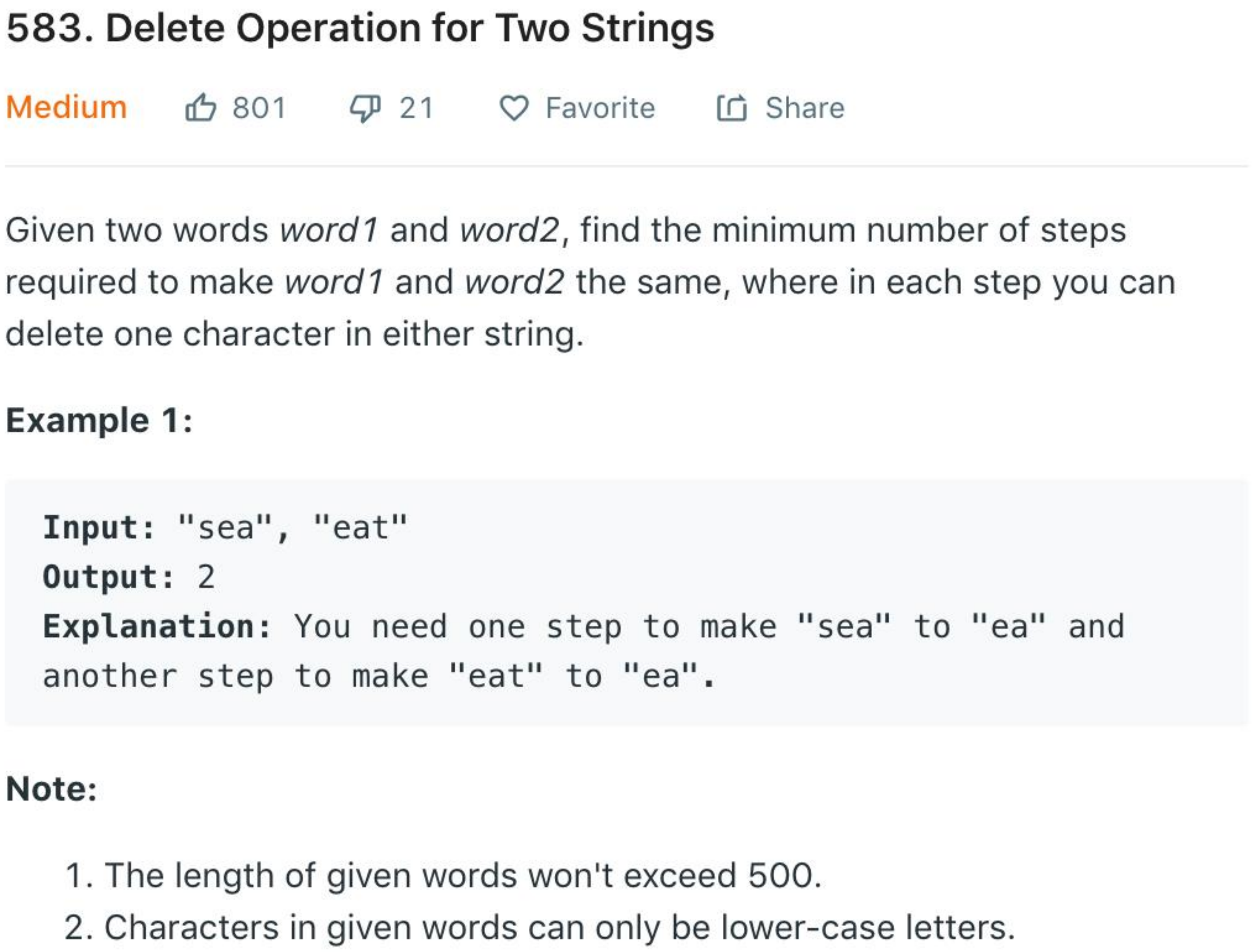
```
Input: [10,9,2,5,3,7,101,18]
Output: 4
Explanation: The longest increasing subsequence is
[2,3,7,101], therefore the length is 4.
```

Note:

- There may be more than one LIS combination, it is only necessary for you to return the length.
- Your algorithm should run in $O(n^2)$ complexity.

Follow up: Could you improve it to $O(n \log n)$ time complexity?

Leetcode583



583. Delete Operation for Two Strings

Medium 👍 801 👎 21 ♡ Favorite ⇱ Share

Given two words *word1* and *word2*, find the minimum number of steps required to make *word1* and *word2* the same, where in each step you can delete one character in either string.

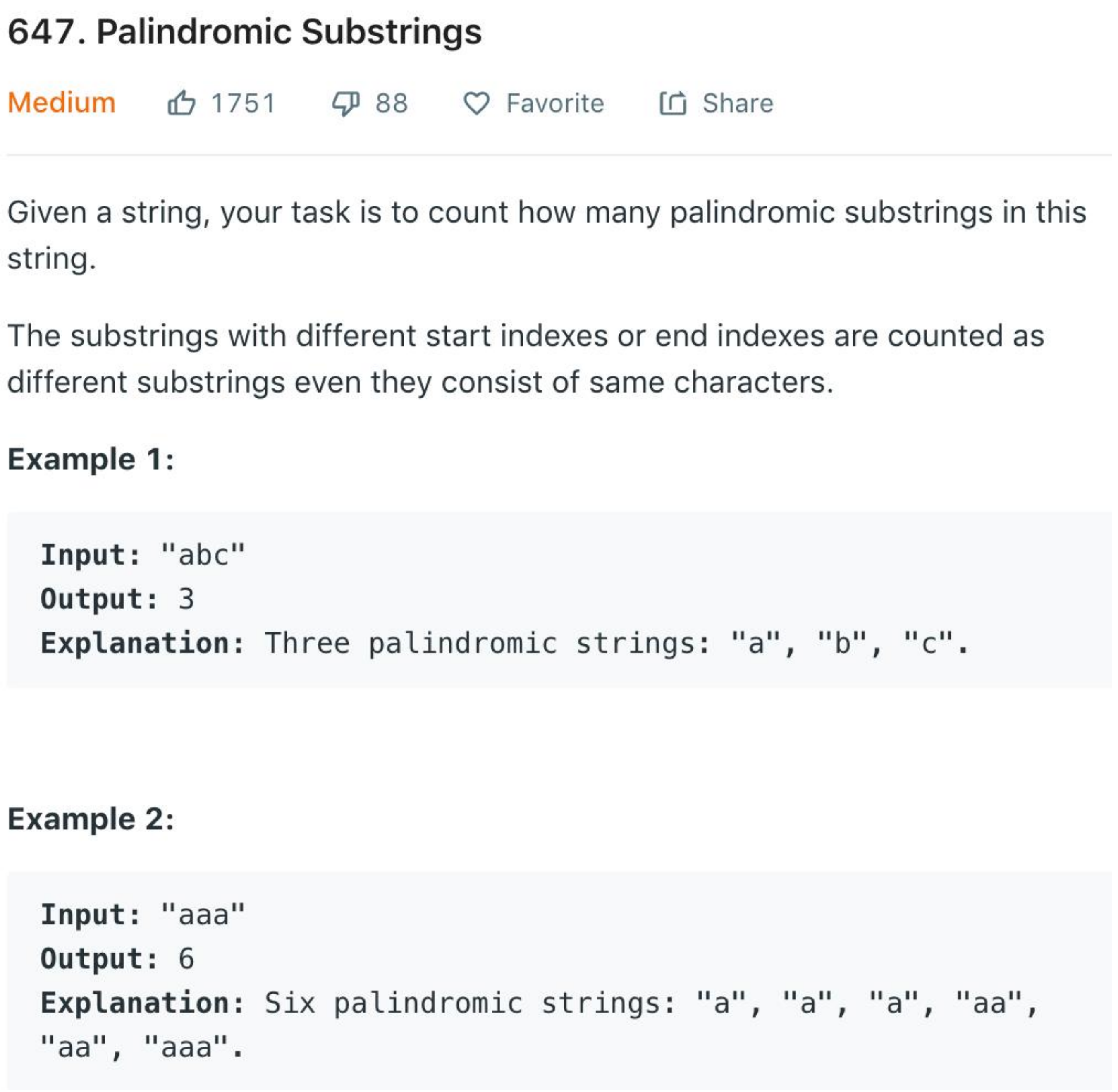
Example 1:

```
Input: "sea", "eat"
Output: 2
Explanation: You need one step to make "sea" to "ea" and
another step to make "eat" to "ea".
```

Note:

1. The length of given words won't exceed 500.
2. Characters in given words can only be lower-case letters.

Leetcode647



647. Palindromic Substrings

Given a string, your task is to count how many palindromic substrings in this string.

The substrings with different start indexes or end indexes are counted as different substrings even they consist of same characters.

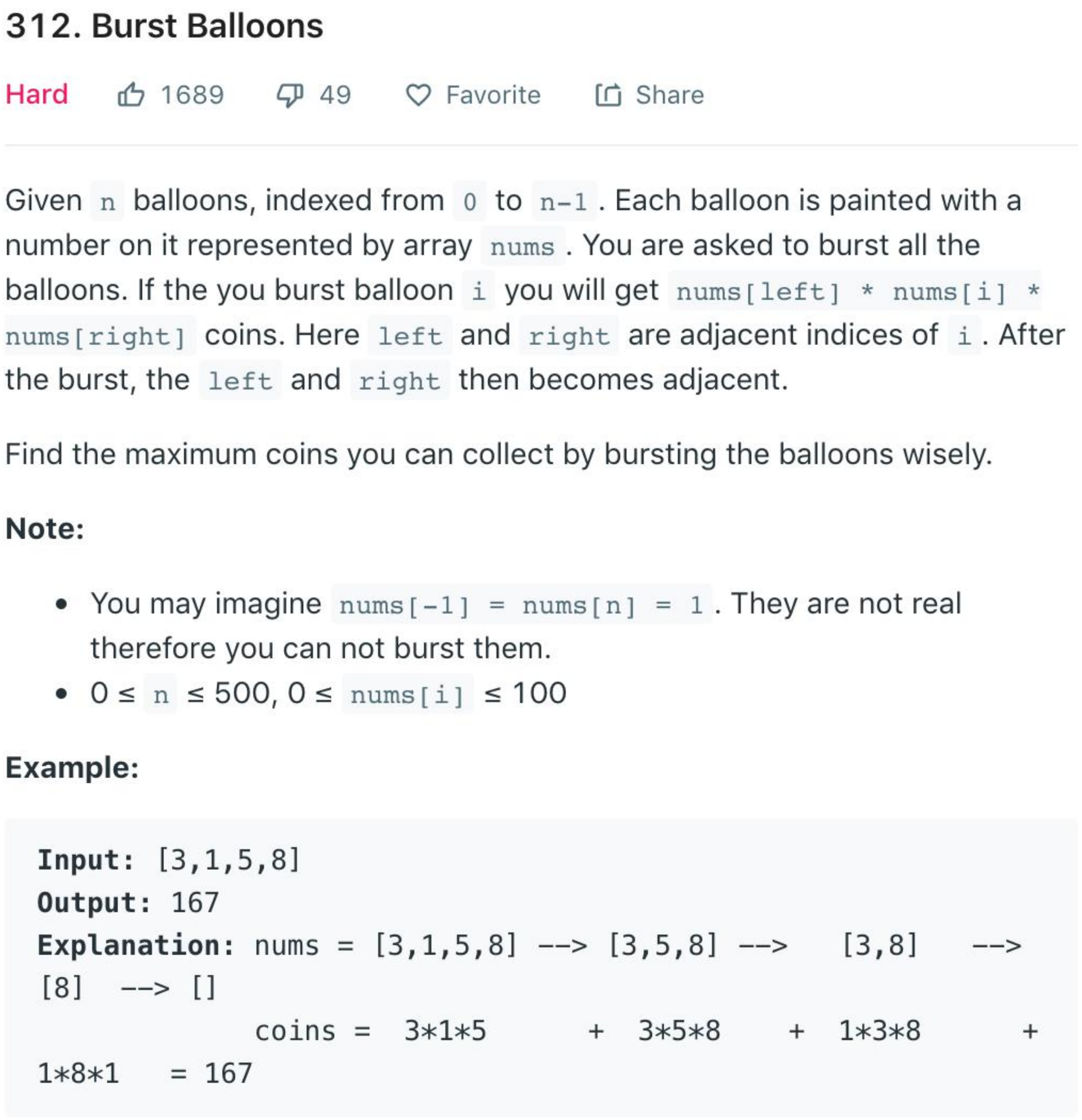
Example 1:

```
Input: "abc"
Output: 3
Explanation: Three palindromic strings: "a", "b", "c".
```

Example 2:

```
Input: "aaa"
Output: 6
Explanation: Six palindromic strings: "a", "a", "a", "aa",
"aa", "aaa".
```

Leetcode312



312. Burst Balloons

Hard 👍 1689 👎 49 ♡ Favorite Share

Given `n` balloons, indexed from `0` to `n-1`. Each balloon is painted with a number on it represented by array `nums`. You are asked to burst all the balloons. If the you burst balloon `i` you will get `nums[left] * nums[i] * nums[right]` coins. Here `left` and `right` are adjacent indices of `i`. After the burst, the `left` and `right` then becomes adjacent.

Find the maximum coins you can collect by bursting the balloons wisely.

Note:

- You may imagine `nums[-1] = nums[n] = 1`. They are not real therefore you can not burst them.
- 0 ≤ `n` ≤ 500, 0 ≤ `nums[i]` ≤ 100

Example:

```
Input: [3,1,5,8]
Output: 167
Explanation: nums = [3,1,5,8] --> [3,5,8] -->   [3,8]   -->
[8]  --> []
             coins =  3*1*5      +  3*5*8    +  1*3*8      +
1*8*1   = 167
```

02背包问题

背包问题通解

1. 0-1 背包
2. 完全背包
3. 多重背包

4. 混合背包
5. 二维费用背包
6. 分组背包

0-1背包

有 N 件物品和一个容量为M的背包。放入第i件物品体积是Ci，得到的价值是Vi。现在我们放入一些物品，问价值总和最大为多少

示例：
M=11
C = [3 4 2 6 1]
V = [5 6 1 7 2]

Ans:
15 (选择物品2 4 5)

0-1背包

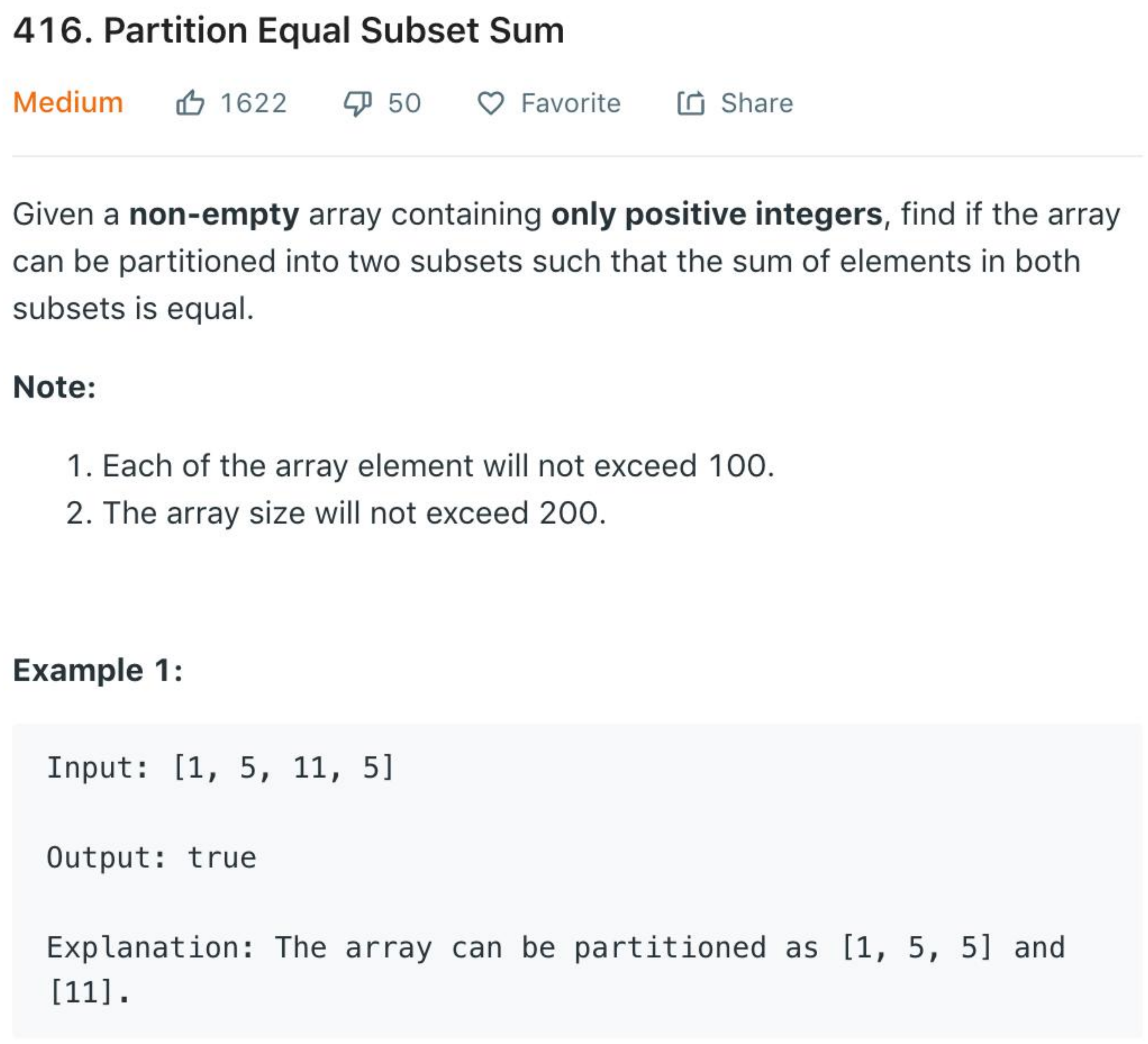
状态定义：
F[i,j] 表示前i 件物品放入一个容量为 j 的背包可以获得的最大价值

状态转移方程:
F[i,j] = max{F[i-1,j],F[i-1,j-Ci]+Vi}

时间复杂度：O(NM)
空间复杂度：O(NM)

空间优化后：O(M)

Leetcode416



416. Partition Equal Subset Sum

Given a **non-empty** array containing **only positive integers**, find if the array can be partitioned into two subsets such that the sum of elements in both subsets is equal.

Note:

1. Each of the array element will not exceed 100.
2. The array size will not exceed 200.

Example 1:

```
Input: [1, 5, 11, 5]

Output: true

Explanation: The array can be partitioned as [1, 5, 5] and
[11].
```

完全背包

有 N 种物品和一个容量为M的背包。放入第i种物品体积是Ci ，得到的价值是Vi。每种物品的数目有无限个，在不超过容量限制的情况下，总价值最大为多少

示例：

M=11

C = [3 4 2 6 1]

V = [5 6 1 7 2]

Ans:

16 (选择2个物品5，1个物品1、4)

完全背包

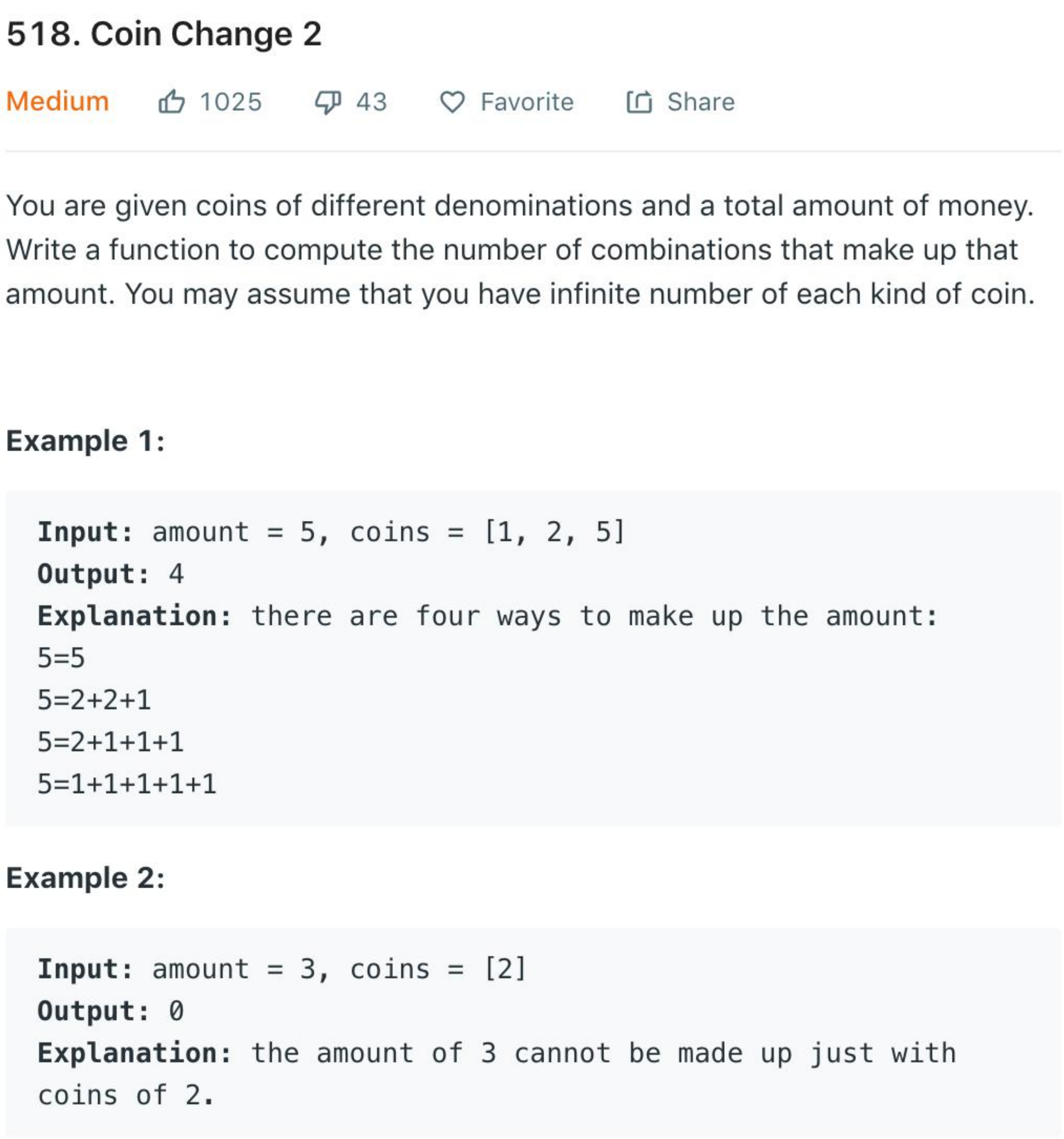
状态定义：
F[i,j] 表示前i 件物品放入一个容量为 j 的背包可以获得的最大价值

状态转移方程:
F[i,j] = max{F[i-1,j],F[i,j-Ci]+Vi}

时间复杂度：O(NM)
空间复杂度：O(NM)

空间优化后：O(M)

Leetcode518



518. Coin Change 2

Medium 👍 1025 👎 43 ♡ Favorite Share

You are given coins of different denominations and a total amount of money. Write a function to compute the number of combinations that make up that amount. You may assume that you have infinite number of each kind of coin.

Example 1:

```
Input: amount = 5, coins = [1, 2, 5]
Output: 4
Explanation: there are four ways to make up the amount:
5=5
5=2+2+1
5=2+1+1+1
5=1+1+1+1+1
```

Example 2:

```
Input: amount = 3, coins = [2]
Output: 0
Explanation: the amount of 3 cannot be made up just with
coins of 2.
```

多重背包

有 N 种物品和一个容量为M的背包，每种物品有一个数量上限Li。第i种物品体积是Ci，得到的价值是Vi。现在我们放入一些物品，问价值总和最大为多少

示例：
M=10
L = [2 2 5 1]
C = [3 4 2 1]
V = [7 5 3 3]

Ans:
20 (选择2个物品一，1 个物品三，1个物品四)

多重背包

状态定义：

$F[i,j]$ 表示前i 件物品放入一个容量为 j 的背包可以获得的最大价值

状态转移方程:

$F[i,j] = \max\{F[i-1,j],F[i-1,j-k*Ci]+k*Vi\}\quad k<=Li \text{ and } k*Ci<=j$

时间复杂度：$O(NM\max(L)) = O(NM^2)$

空间复杂度：$O(NM)$

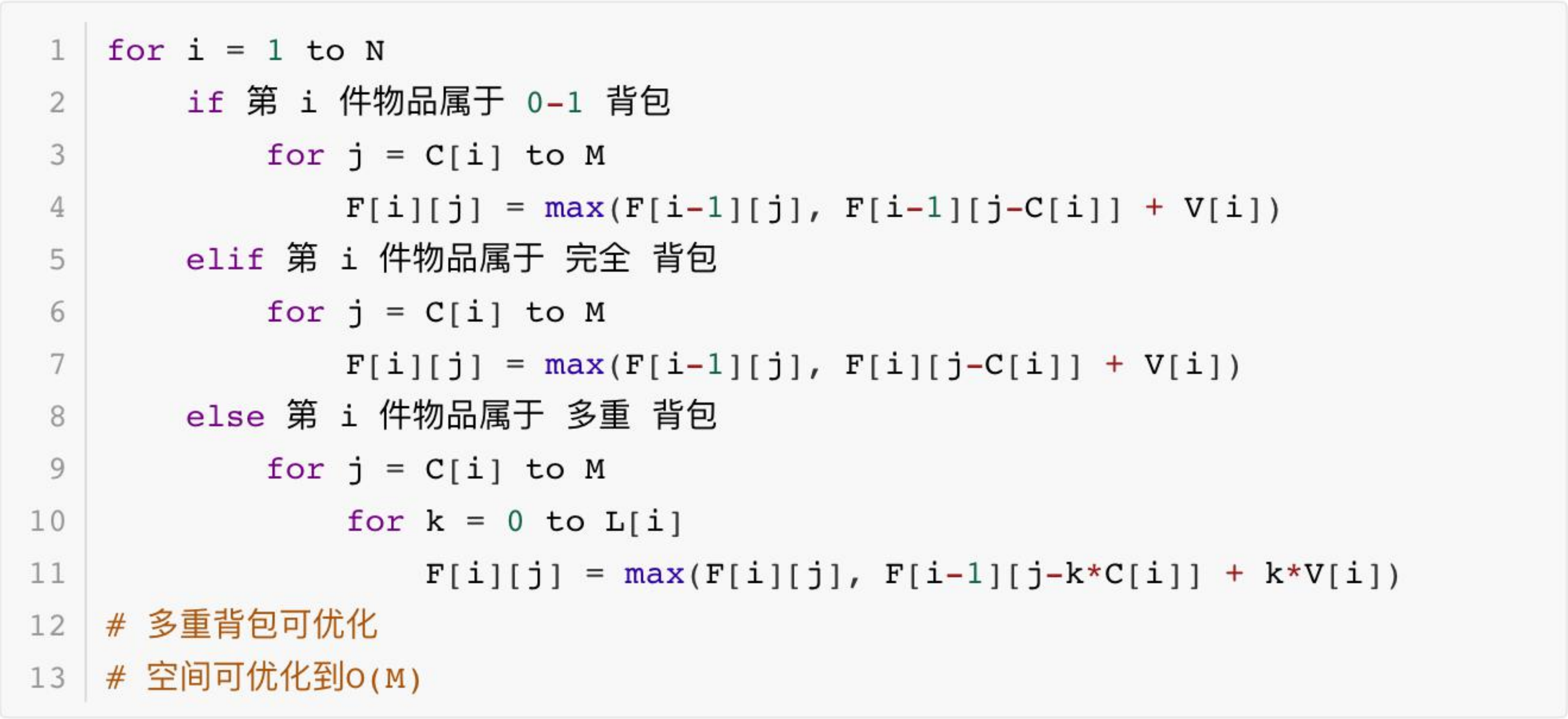
二进制优化后：

时间复杂度：$O(NM\log(\max(L)))=O(NM\log M)$

空间复杂度：$O(M)$

混合背包

将前面三种背包问题混合起来。也就是说，有的物品只可以取一次（01 背包），有的物品可以取无限次（完全背包），有的物品可以取有限次(多重背包），问价值总和最大为多少



```
for i = 1 to N
    if 第 i 件物品属于 0-1 背包
        for j = C[i] to M
            F[i][j] = max(F[i-1][j], F[i-1][j-C[i]] + V[i])
    elif 第 i 件物品属于 完全 背包
        for j = C[i] to M
            F[i][j] = max(F[i-1][j], F[i][j-C[i]] + V[i])
    else 第 i 件物品属于 多重 背包
        for j = C[i] to M
            for k = 0 to L[i]
                F[i][j] = max(F[i][j], F[i-1][j-k*C[i]] + k*V[i])
# 多重背包可优化
# 空间可优化到O(M)
```

二维费用背包

有N件物品，每件物品有两个费用，第i 件物品所需的两种费用分别为Ci 和Di。两种费用可付出的最大值（也即两种背包的容量）分别为V 和U。物品的价值为Wi。问价值总和最大为多少

费用加了一维，只需状态也加一维即可。设F[i,v,u] 表示前 i 件物品付出两种费用分别为 v 和 u 时可获得的最大价值。

状态转移方程就是:

F[i,v,u] = max{F[i-1,v,u],F[i-1,v-Ci,u-Di] + Wi}

分组背包

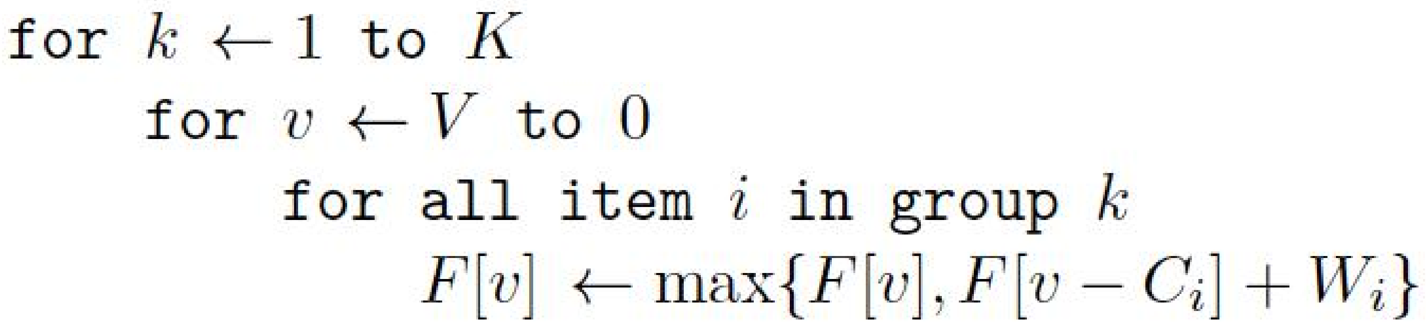
有N 件物品和一个容量为 M 的背包。第i 件物品的费用是Ci，价值是Vi。这些物品被划分为K 组，每组中的物品互相冲突，所以每组最多只能选一个物品。求总价值最大为多少

F[k,j] 表示前k 组物品花费费用 j 能取得的最大价值和

$F[k,j] = \max\{F[k-1,j], F[k-1,j-C_i] + V_i \quad | \quad \text{item } i \text{ in group } k\}$

分组背包

有N 件物品和一个容量为 M 的背包。第i 件物品的费用是Ci，价值是Vi。这些物品被划分为K 组，每组中的物品互相冲突，所以每组最多只能选一个物品。求总价值最大为多少



```
for $k \leftarrow 1$ to $K$
    for $v \leftarrow V$ to $0$
        for all item $i$ in group $k$
            $F[v] \leftarrow \max\{F[v], F[v - C_i] + W_i\}$
```