# Zoom Link: <https://us05web.zoom.us/j/85198478429?pwd=YytjNkxVSFRXUXhEd0d5YUFpV1BSQT09>

google 面经：

2021的：

<https://docs.google.com/document/d/1ubIZGQx3v1TbpAEL38Je8XzUJZm7krzmochz3Emw0uE/edit>

之前的：

<https://docs.google.com/document/d/1c_lF3DxgJm_LgL-kFf5kAtdAxwfnvnFa4_89HwsVdl0/edit>

GOOGLE

Rolling Hash + prefix sum +state compress（状态压缩）啥题呀？？？

# 6.22

LC 140

class Solution(object):

def wordBreak(self, s, wordDict):

"""

:type s: str

:type wordDict: List[str]

:rtype: List[str]

"""

memo = {len(s): ['']}

def sentences(i):

if i not in memo:

memo[i] = []

for j in range(i+1, len(s)+1):

if s[i:j] in wordDict:

for tail in sentences(j):

if tail != '':

memo[i].append( s[i:j] + ' ' + tail)

else:

memo[i].append( s[i:j] )

return memo[i]

return sentences(0)

1 1 1 1 2 3

i

public List<List<Integer>> combinationSum2(int[] nums, int target) {

List<List<Integer>> res = new ArrayList<>();

if(nums==null || nums.length == 0){

return res;

}

Arrays.sort(nums);

helper(nums, target, new ArrayList<Integer>(), 0, res);

return res;

}

private void helper(int[] nums, int remain, ArrayList<Integer> path, int idx, List<List<Integer>> res) {

if(remain == 0){

res.add(new ArrayList<>(path));// 必须重新new 一 }

for(int i = idx; i<nums.length; i++){

if(nums[i] > remain){

continue;

}

if(i > idx && nums[i] == nums[i-1]){

continue;

}

path.add(nums[i]);

helper(nums, remain - nums[i], path, i+1, res);// 不应该是i++

path.remove(path.size() - 1);

}

}

1 3 2 1 1 1 target 5

base

path.add(idx) = [1]

dfs(idx + 1, path)

# 6.23

1. word Ladder 存图的问题，why dfs better
2. sort algorithm：结合 linked list：147 148

Selection Sort

Unstable

Why？

Ex. [(7),3,{7},1]

idx=0

Find min Value = 1

Swap 1, (7)

List = [1,3,{7},(7)]

A b c d …. g

Count/Buckets Sort

1 3 5 2 2 4 3 5 1 2

次数 Bucket sort：前面换成buckets

count1 2 1 1

count2 3 2 2 2

count3 2 3 3

count4 1 4

count5 2 5

int[]

B+ tree -> Radix sort

123 254 120

9 8 7 … 0

123 254 120

123 120 254

120 123 254

Heap sort : stable?????

Consider array 21 20a 20b 20c(already in max-heap format)

20c

20a 20b

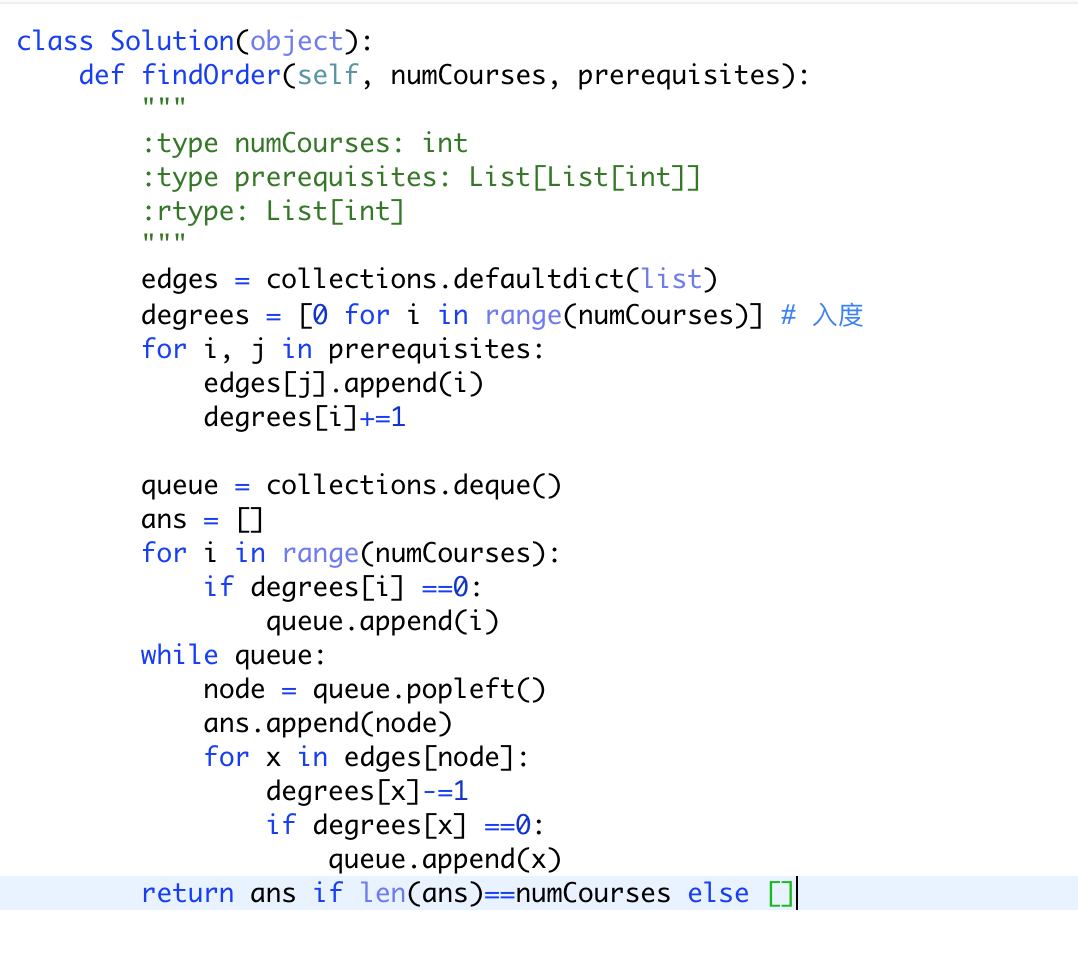
20c 20a 20b

GRAPH:

Lc133: Use Map/HashMap to keep original node with its copy. Use DFS to create the graph

Lc207: 拓补排序。利用课程关系建图。记录visit标记。dfs遍历每个点，**只要存在环**，return false.

Lc210:利用课程关系建图。 记录每个node的入度。把入度为0的node放入queue中，遍历queue。输出ans。



LC269. Alien Dictionary 于lc210类似。难点在于建图

Lc323. union -find. 构建一个DSU，以及union和find方法。如果u,v之间存在一条边，则call union(u,v).最后count u.find()的长度

when to use？？？

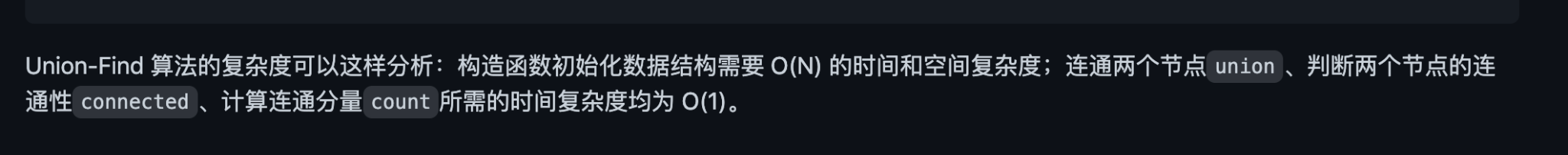
Disjoint set union

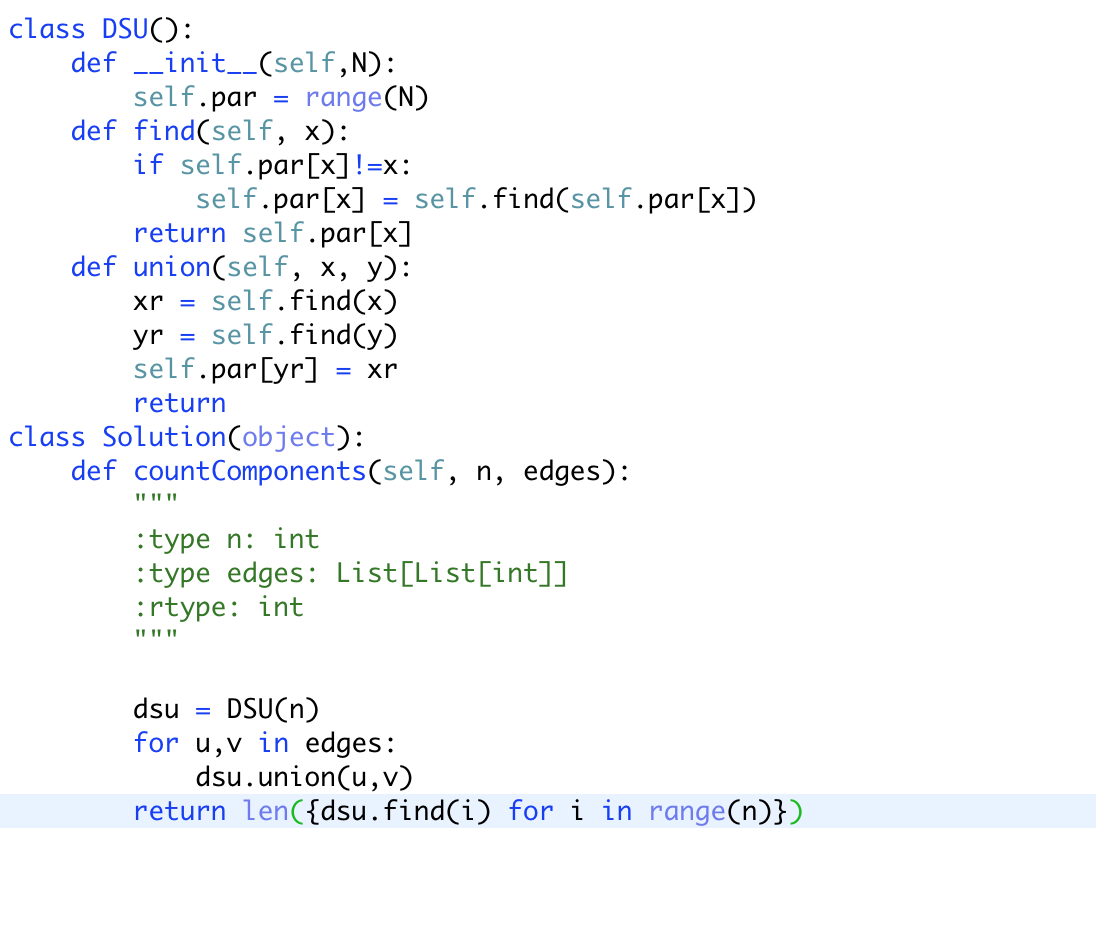
p3 -> p4 ->p5

p1 -> p2 ->

p ->

Complexity: https://github.com/labuladong/fucking-algorithm/blob/master/%E7%AE%97%E6%B3%95%E6%80%9D%E7%BB%B4%E7%B3%BB%E5%88%97/UnionFind%E7%AE%97%E6%B3%95%E8%AF%A6%E8%A7%A3.md#:~:text=Union%2DFind%20%E7%AE%97%E6%B3%95%E7%9A%84%E5%A4%8D%E6%9D%82,%E5%9D%87%E4%B8%BAO(1)%E3%80%82





lc332.HashMap建图。 DFS遍历，并输出ans

Lc1192. Tarjan’s Algorithm.

# 6.29

This Week:

Graph

Two Pointer/Sliding Window

数据结构的相互实现：没整理

linked list 实现 queue

array 实现 Arraylist

dequeue/queue <-> stack

OOD:没复习

Concurrency & Parallel:

Synchronized/ Lock

# 7.5

小曾的任务：

简历弄好

小杨的任务：

简历弄好

* 数据结构的相互实现
* DP

Graph

最短路径：

单源： Dijkstra

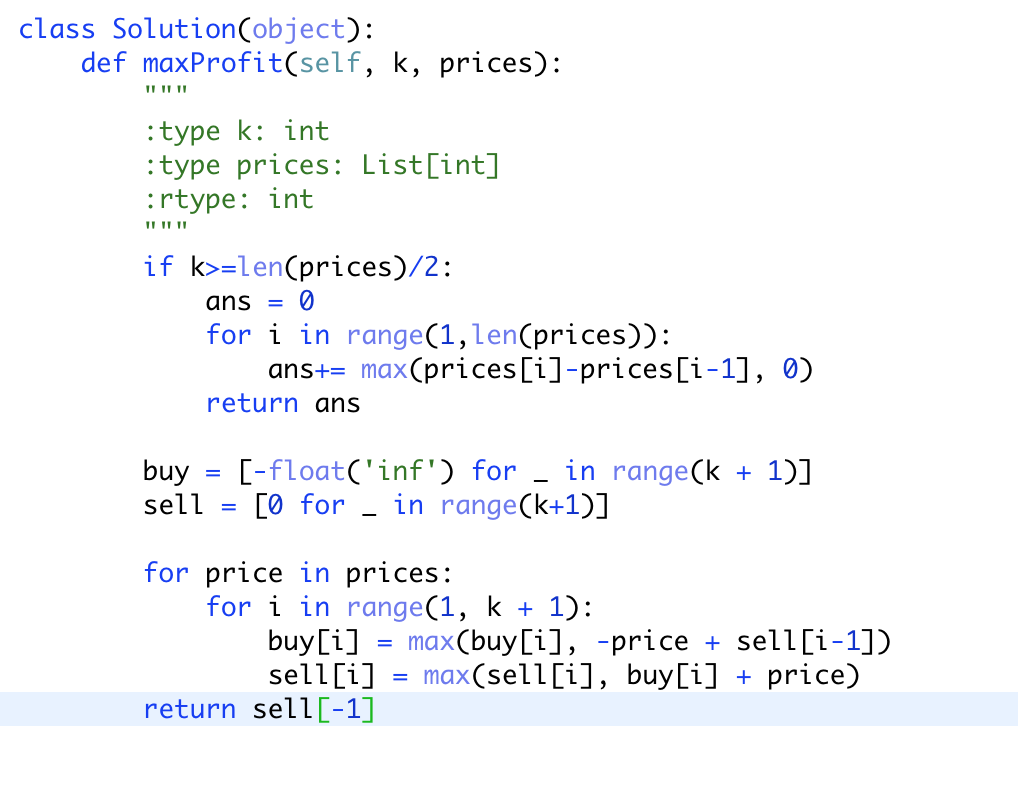
最小生成树：

Prim: 选一条最小边连接当前点集合

Kruskal: 每次选最小边，if此边的两个vertex不属于同一个集合，则连接两个集合，否则continue

Left Topics:

1. Interval (meeting room, breaking tie)
2. bit
3. Search , find K th
4. Trie
5. Stack
6. Greedy



# 7.18

Lc 1937. Maximum Number of Points with Cost dp from O(m\*n\*n) to O(m\*n)

Lc1938. Trie

# 7.23

Lc 1648 heap+ binarySearch

# 7.29

Lc1760 binarySearch

Lc1567 medium dp

Lc1801 heap

Lc1614 ez

# 7.30:

Rope Cut:

https://leetcode.com/problems/minimum-cost-to-cut-a-stick/

155: Min Stack

Word Break II: DFS + Trie的做法！

# 8月schedule：

小曾还要复习亚麻黄金！

1. Java OOD system Design陪练
2. OOD加强

什么file filter

1. System Design 宋杨 至少看完tiny url
2. 简历的项目细节

# 8.1:

Frog Jump 的DP做法。

# 8.2:

1.Alien Dictionary: build graph有没有简单一点的做法呢？

2. 给一堆（员工，经理）的数组，让找出每个经理管理员工的数量。例如：输入是（A, B）(C, B) (B, D)，结果就是（A, 0）(B, 2) (C, 0) (D, 3) —> Graph (建图 + b/d fs) （onsite 面经

D的数量 = degree[D] + degree[B]

2. Trap Rain Water I && II 407

# 8.3:

FaceBook Onsite: -> BFS 带权重

1. kth smallest number in f(x,y,z) = a^x \* b^y \* c^z (x,y,z>=0, a,b,c > 1) (facebook onsite)

需要Override hashcode() 和 equals()

hashcode 和equals需要同步改写

https://www.geeksforgeeks.org/override-equalsobject-hashcode-method/

Hints:

把(x,y,z)从(0,0,0)到(k-1,k-1,k-1)共3K个结果放进heap=>排序=>取出第k个。T:O(klogk) tuple(f, func(根据题意具体的排序方法))

1. LC318 -> 用bit去查两个string 是否相等！！太聪明了
2. Longest Ascending subsequence!!!

S1: dp[i] represents the longest ascending subsequence length including/ending with array[i]

dp[i] = max(dp[k]) + 1, for all the k < i && array[k] < array[i]

Time O(n^2) Space O(n)

Mono stack？？？？

S2: O(nlogn) + binary Search！

**2 1 4 5 3 7 6 4 5**

length 1 2 3 4

value 1 3 4 5

arraylist：index 代表长度 from 1

Value：相同长度下最小的值

# 8.4

Lc1941-1944:

1944. Number of Visible People in a Queue

mono stack

单调栈，根据遍历方向不同，有两种思路

1943. Describe the Painting

preSum

1942. The Number of the Smallest Unoccupied Chair

Heap

1941 ez

1. Longest Common Subsequence：

DP怎么优化？？？

# 8.5

可怕

LC 1869-1873

1.[1869. Longer Contiguous Segments](https://leetcode.com/problems/longer-contiguous-segments-of-ones-than-zeros) 给一个string由若干0和1组成，问最长连续的1是否长于最长连续的0

2.

| [1870. Minimum Speed to Arrive on Time](https://leetcode.com/problems/minimum-speed-to-arrive-on-time) |  |
| --- | --- |

一个binary Search EZ

3. 1871. Jump Game VII

Dp[i]=true -> 表示pos i 可以jump到。

Si ==’0’ && 在[i-max, i-min]中存在j 使得dp[j]==true

4. 1872. Stone Game VIII

preSum + dp 写出两层forloop 然后简化成一层 T:O(n)

# 

# 8.6

用stack 来实现 pre，in，post order的遍历，非常重要！

Lc 1946-1948

[1946. Largest Number After Mutating](https://leetcode.com/problems/largest-number-after-mutating-substring) EZ GREEDY

[1947. Maximum Compatibility Score Sum](https://leetcode.com/problems/maximum-compatibility-score-sum) DFS 暴力解

[1948. Delete Duplicate Folders in Sys](https://leetcode.com/problems/delete-duplicate-folders-in-system) Trie+ Hash

# 8.7

1504. Count Submatrices With All Ones : 转换得很巧妙！

# 8.11

LC815

LC115 有bug why

Q3 Given a String, first non-repeating character / first repeating / second non-repeating / second repeating …

Q4 Stream flow

Double Linked List: Used to keep the order: keep a list of nodes that appear by first occurrence

hashmap<key, node> when it has key, but node reference is null, means this node has shown before, we don’t need to add it to the map and double linked list again! then we can just

# 8.13

1000 和1547好像还没怎么吃透。。。

# 8.14

[1959. Minimum Total Space Wasted With K Resizing Operations](https://leetcode.com/problems/minimum-total-space-wasted-with-k-resizing-operations) 好题！DP+想办法降复杂度

[1960. Maximum Product of the Length of Two Palindromic Substrings](https://leetcode.com/problems/maximum-product-of-the-length-of-two-palindromic-substrings) 马拉车算法+回文

# 8.15

Real count sort: 用prefix sum的方式来做，奇妙

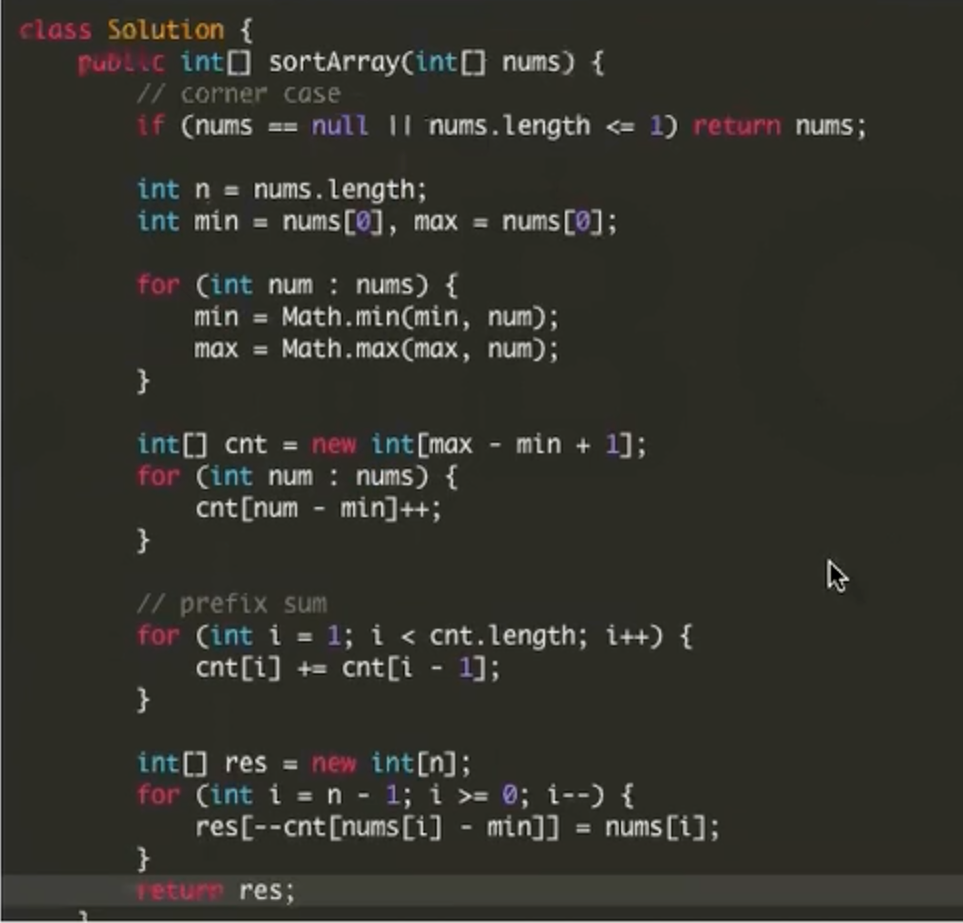
nums = [5,1,1,2,0,0]

0:2 1：2 2:1 5:1

0 1 2 3 4 5

prefix Sum 2 4 5 5 5 6

sum的是count的累积和



1000, 877, 1872

LC 318: bit 优化，hashset如何优化

LC42

Move Zero 从右忘左填

# 8.17:

Session 5

[1653. Minimum Deletions to Make String Balanced](https://leetcode.com/problems/minimum-deletions-to-make-string-balanced) DP

[1654. Minimum Jumps to Reach Home](https://leetcode.com/problems/minimum-jumps-to-reach-home) 难点：确定dp(i)中 i参数的范围 裴蜀定理 ax+by=c 如果此等式有整数解，那么c一定是gcd(a,b)的倍数，即c%gcd(a,b)==0

推论： ax+by=1有正整数解 的充要条件是 a,b互质， 即gcd(a,b)==1

[1655. Distribute Repeating Integers](https://leetcode.com/problems/distribute-repeating-integers) 状态压缩DP+子集枚举

1655. dp(i,j ) freq[i] 能不能满足 j这个集合 customer的需求

[1,10] m

求dp(len(freq),2^m-1)

j: 【0, 1023】

j: bit 的形式

512 0100000000

j: 0100100010

0100100001

S : 0100100000

dp(i,j) freq[i]满足j所代表的这个子集

Freq = [1,5,4,3,2]

Idx 0 1 2 3 4

C1 2

C2 1

S 是j的子集

， dp(i-1, j-s)==true

For (int s=j; s!=0; s = ((s-1)&j)

0 - 1023 代表集合的个数

512 0100000000

j: 0100100010

s1 0100100001

S2 : 01000111111

0100100010

# 8.18:

LC315 : 分治 + Binary Search

[1691. Maximum Height by Stacking Cuboids](https://leetcode.com/problems/maximum-height-by-stacking-cuboids) DP+处理相同长宽高方块

# 8.19:

KMP algorithm

1686

# 8.21:

KMP 代码

