DFS- Depth First Search

深度优先搜索 - 递归

代码魔术师
magiciendecode.fr

DFS

1. Permutation
2. Combination & Backtracking
3. Graph (Tree & Grid)
4. Union Find
5. Other

代码魔术师
magiciendecode.fr

Recursion + Memo && Backtracking

Recursion + Memorization

```
dfs ( state, memo) {

  if ( memo.contains(state) ){
      return it
  }

  calculate current state using dfs(nextState,memo)

  update memo
  return current result
}
```

Backtracking

```
dfs ( state ){
  if ( end condition ){
     handle  result
  }else{
     foreach possible next state:
        1. change state to state2
        2.  dfs( state2 )
        3. reset state2 to state
  }
}
```

代码魔术师
magiciendecode.fr

77. Combinations

https://leetcode.com/problems/combinations/

排列 Permutation (Arrangement)

$$P_n^m = \boxed{\frac{n!}{(n-m)!}} = n(n-1)(n-2)\ldots\ldots(n-m+1)$$

n until n-m+1

1 2 3 => 1 2
2 1
1 3
3 1
2 3
3 2

P3(2) = 3 * 2

组合 Combinaiton

$$C_n^m = \boxed{\frac{n!}{(n-m)!m!}} = \underbrace{\frac{n(n-1)(n-2)\ldots\ldots(n-m+1)}{m(m-1)(m-2)\times\ldots\ldots\times 1}}$$

n until m

m until 1

1 2 3 => 1 2
1 3
2 3

C3(2) = 3 * 2 / 2 = 3

代码魔术师
magiciendecode.fr

78. Subsets & 90. Subsets II

https://leetcode.com/problems/subsets/

https://leetcode.com/problems/subsets-ii/

[1, 2, 3]

1

1 2

1 2 3

2

2 3

3

[1, 2, 2,]

1 ==> 2

~~1 ==> 2~~

> startIndex

>0

==

contunue

代码魔术师
magiciendecode.fr

797. All Paths From Source to Target

https://leetcode.com/problems/all-paths-from-source-to-target/

```
fun allPathsSourceTarget(graph: Array<IntArray>): List<List<Int>> {
    val result: MutableList<MutableList<Int>> = LinkedList()
    val subsets: MutableList<Int> = LinkedList()

    dfs(result, subsets, graph, 0)
    return result
}
```

```
subsets.add(index)
if (graph[index].isEmpty()) {
    result.add(ArrayList(subsets))
} else {
    graph[index].forEach {
        dfs(result, subsets, graph, it)
        subsets.removeAt(subsets.size - 1)
    }
}
```

1. deep copy result

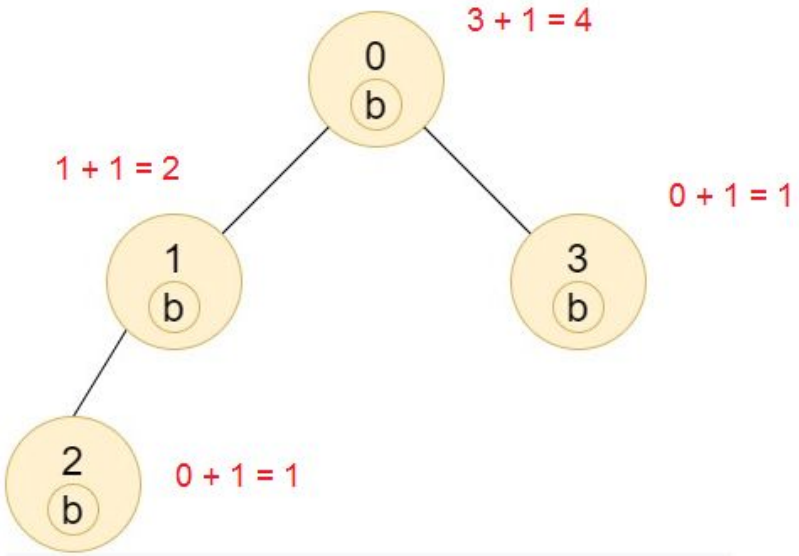
2. end condition ?

3. backtrace

代码魔术师
magiciendecode.fr

1519. Number of Nodes in the Sub-Tree With the Same Label

https://leetcode.com/problems/number-of-nodes-in-the-sub-tree-with-the-same-label/




```
Input: n = 4, edges = [[0,1],[1,2],[0,3]], labels = "bbbb"
Output: [4,2,1,1]
```

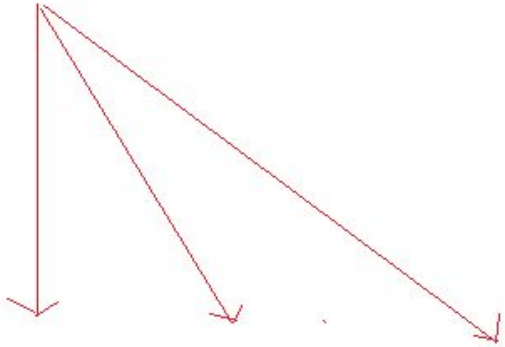
```
val currentIndex = labels[current] - 'a'
++value[currentIndex]
val currentValue = value[currentIndex]
graph[current]?.forEach {
    if (it != previous) {
        dfs(graph, result, value, it, current, labels)
    }
}
result[current] = value[currentIndex] - currentValue + 1
```

代码魔术师
magiciendecode.fr

216. Combination Sum III

https://leetcode.com/problems/combination-sum-iii/

K = 3, N = 9



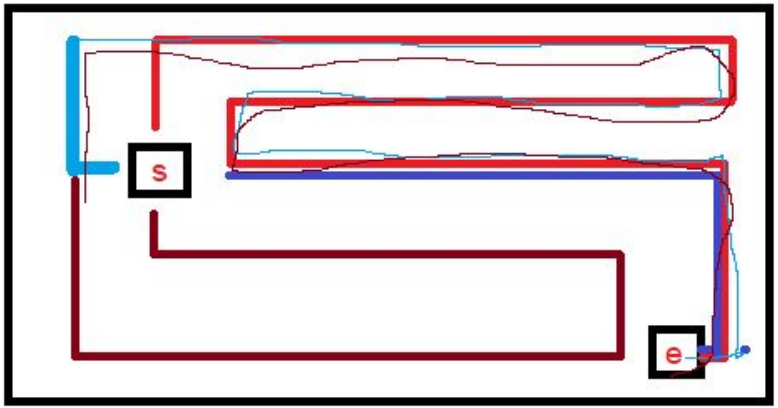
[1 2 3 4 5 6 7 8 9]	[1 2 3 4 5 6 7 8 9]	[1 2 3 4 5 6 7 8 9]

```
private void dfs(
        int k,
        int remain,
        int start,
        List<Integer> subsets,
        List<List<Integer>> results
) {
    // end condition
    if (remain == 0 && k == subsets.size()) {
        results.add(new ArrayList<>(subsets));
    }
    // not valid results, so return
    if (remain < 0 || k == subsets.size()) {
        return;
    }
    // backtracking
    for (int i = start; i <= 9; i++) {
        subsets.add(i);
        dfs(k, remain - i, i + 1, subsets, results);
        subsets.remove(subsets.size() - 1);
    }
}
```

代码魔术师
magiciendecode.fr

329. Longest Increasing Path in a Matrix

https://leetcode.com/problems/longest-increasing-path-in-a-matrix/




```
public int longestIncreasingPath(int[][] matrix) {
    int max = 0;
    for (int i = 0; i < matrix.length; i++) {
        for (int j = 0; j < matrix[0].length; j++) {
            max = Math.max(max, dfs(matrix, i, j));
        }
    }
    return max;
}
```

Time Limit Exceed => DP

how to avoid redundant calculation ? Dynamic Programming

if dp[][] != 0, take result
else dfs(...)

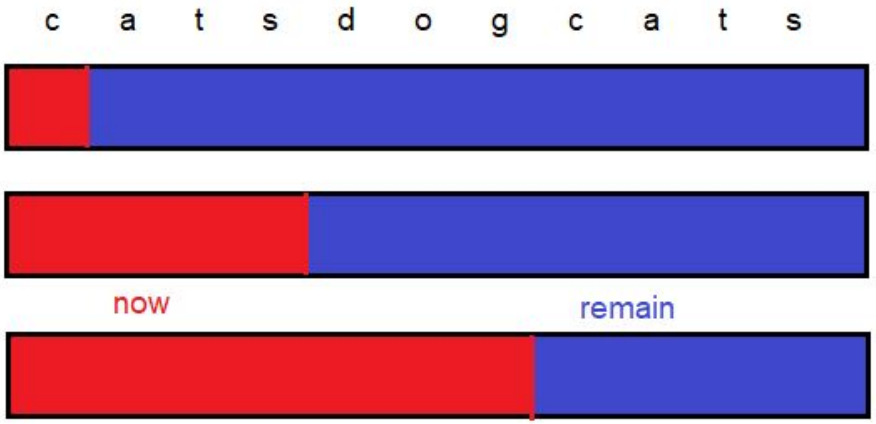
can NOT use breadth first search,

because we should use depth first search to backtract the result.

代码魔术师
magiciendecode.fr

472. Concatenated Words

https://leetcode.com/problems/concatenated-words/




```
if (set.contains(now)) {
    final String remain = word.substring(now.length());
    if (set.contains(remain) || dfs(remain, set)) {
        return true;
    }
}
```

139 Word Break
140 Word BreakII

代码魔术师
magiciendecode.fr

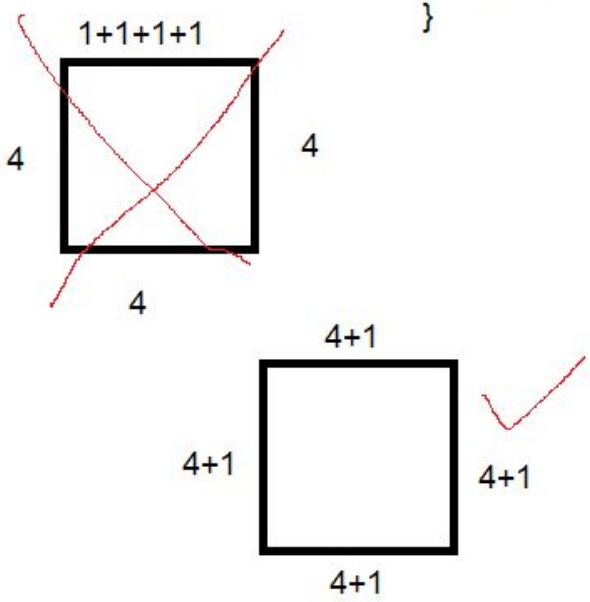
473. Matchsticks to Square

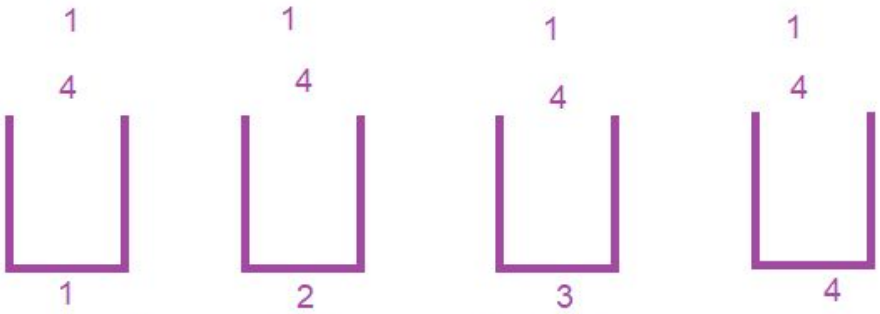
https://leetcode.com/problems/matchsticks-to-square/

[1 1 1 1 4 4 4 4]

```
final int sideLength = sum >> 2;
if (sideLength * 4 != sum) {
    return false;
}
```

1. sort array
2. from large to small, try to construct 4 sides





if there are 3 sides are valid, return true

代码魔术师
magiciendecode.fr

488. Zuma Game

https://leetcode.com/problems/zuma-game/

```
int dfs(String s,hand){
    newS = s.remove3SameChar
    if(newS == "$"){
        0
    }
    if there is possiblity to construct 3 same char
    newHand
    result = min (result, dfs(afterRemove,newHand))
    rollbackHand

    return result
}
```

this algo is try to eliminate single or two chars in board!
it doesn't work for RRWWRRBBRR,WB

RRWWRRBBRR WB

add W after two WW
~~RRWWWRR~~ BBRR
BBRR hand=B

Return -1 ?

RRWWRRBBRR
add W
RRWWRRBBRWR
add B
RRWWRRBBBRWR
resolved

solution:
for each position,
add each char in hand

if not visited,
then dfs to search!

代码魔术师
magiciendecode.fr

488. Zuma Game - Breadth First Search

https://leetcode.com/problems/zuma-game/

RRWWRRBBRR WB

add W after two WW

RRWWWRR BBRR

BBRR hand=B

Return -1 ?

RRWWRRBBRR

add W

RRWWRRBBRWR

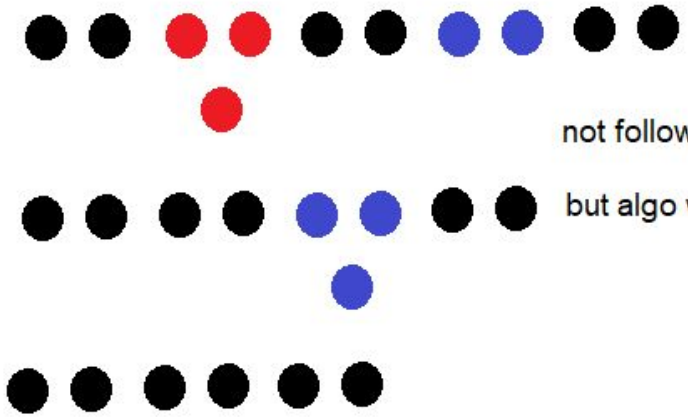
add B

RRWWRRBBBRWR

resolved

solution:
for each position,
add each char in hand

if not visited,
then dfs to search!



not follow the rule of zuma

but algo works

algo return 2
but result should be -1

so Breadth First Search

代码魔术师
magiciendecode.fr

491. Increasing Subsequences

https://leetcode.com/problems/increasing-subsequences/

subsets

[1 2 3]

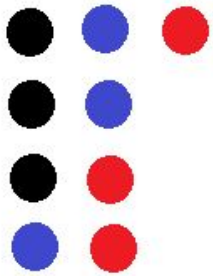
for 1
1 2
1 2 3
1 3

subsets II

[1 2 2 3]

1 2 2 3
1 2 3
~~1 2 3~~

[1 2 3 4 1 1]
1 2 3 4
...



```
dfs (nums, startIndex, results, subsets){
    // clone, deep copy
    results.add(new ArrayList(subset));
    for (int i = startIndex; i < nums.length; i++) {
        subset.add(nums[i]);
        dfs(nums, i + 1, subset, results);
        subset.remove(subset.size() - 1);
    }
}
```

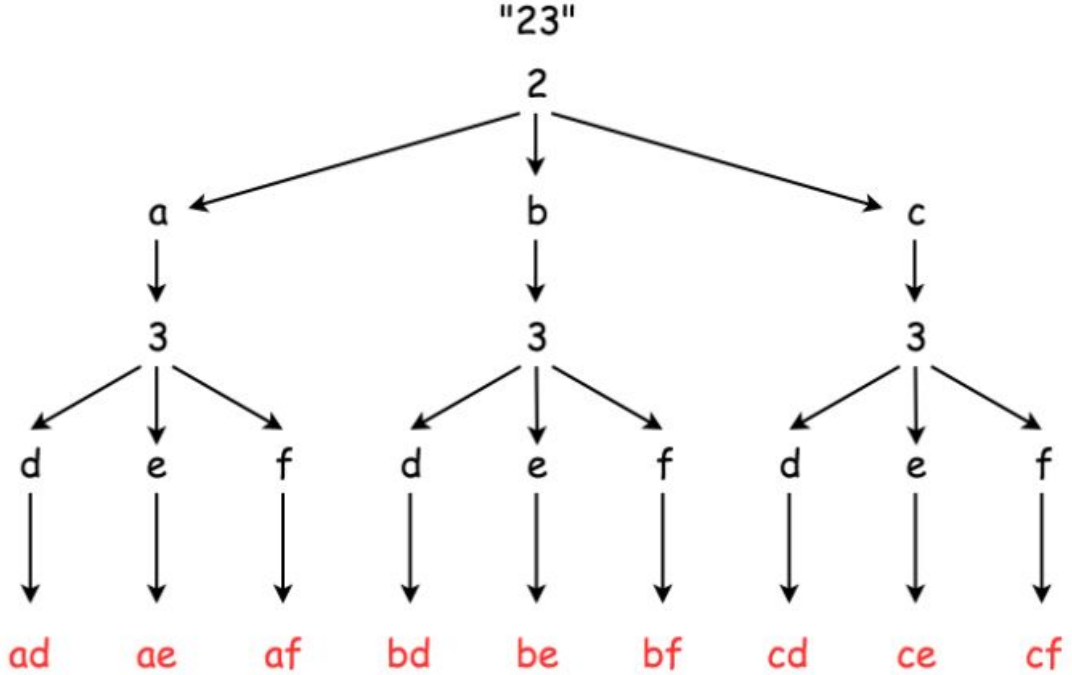
```
if (!subsets.isEmpty() && subsets.get(subsets.size() - 1) > nums[i]
    || visited.contains(nums[i])) {
    continue;
}
```

```
if (i > 0 && i > startIndex && nums[i] == nums[i - 1]) {
        continue;
}
```

代码魔术师
magiciendecode.fr

17. Letter Combinations of a Phone Number

https://leetcode.com/problems/letter-combinations-of-a-phone-number/



output = ["ad", "ae", "af", "bd", "be", "bf", "cd", "ce", "cf"]

"23"

level 1: [a b c]
level 2: [ad ae af bd be bf cd ce cf]

...

functional programming

String "2" ==> [a b c]
String "3" ==> [d e f]
...

[a b c] outer join [d e f]
outer join ...
outer join ...

代码魔术师
magiciendecode.fr

494. Target Sum

https://leetcode.com/problems/target-sum/

[1 1 1 1 1]

each 1 has two cases, +1 or -1.

dfs (index,currentSum)

currentSum += [index]
dfs(index+1,currentSum)

currentSum -= [index]
currentSum -= [index]
dfs(index+1,currentSum)

```
fun findTargetSumWays(nums: IntArray, sum: Int): Int =
    nums.map { listOf(it, -it) }.reduce { acc, list ->
        acc.flatMap { accItem -> list.map { listItem -> accItem + listItem } }
    }.count { it == sum }
```

It is the same logic as phone number to characters.

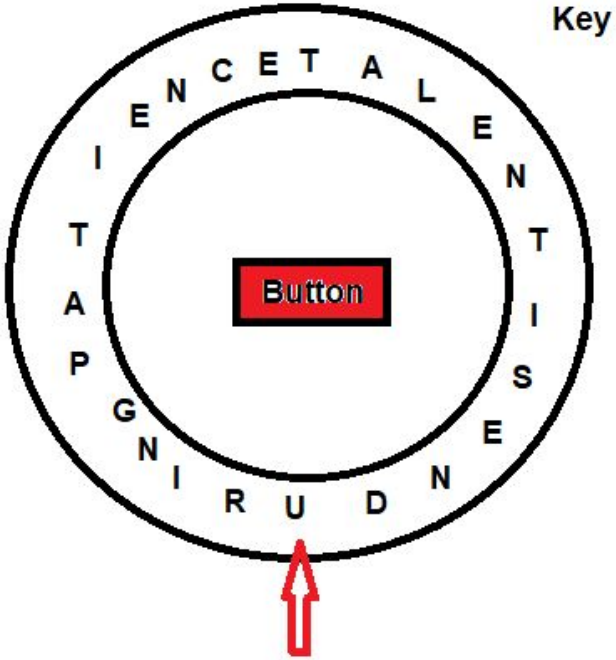
but for phone number, we have input length max = 9 and mapLength = 3.
So max result is pow(3,9)

for this problem, max = 20 and mapLength = 2, max result is pow(2,20)
Time Limit Exceeded.

代码魔术师
magiciendecode.fr

514. Freedom Trail

https://leetcode.com/problems/freedom-trail/



Key = ANTLENTSIDUR

1. for a given index of ring, the minimum steps to get the index of key and + 1 for button.

2. so our depth first search function is dfs(indexRing,indexKey)

3. clockwise and antiClockwise to find the char at key, the dfs, return the minimum of (clock,antiClock)

Time Limit Exceeded
Improvment 1, to find the keyChar, do not need to iterate all ring, for each direction, just iterate half of ring and dfs to find minimum.

Time Limut Exceeded
Improvement 2, as we always compare dfs solution between clockwise and antiClockwise, if for a give indexRing, indexKey, we have calculated the minimal before, just use it.
So maintain a dp[index of ring][index of key] to store the result.

代码魔术师
magiciendecode.fr

312. Burst Balloons

https://leetcode.com/problems/burst-balloons/

1 3 1 5 8 1

3	30	159	167
0	15	135	159
0	0	40	48
0	0	0	40

length = 1

3 15 40 40

length = 2

[3,1] = 15 + 15 = 30

1 3

[1,5] = 135

5 -> 1 = 40 + 24 = 64

1 -> 5 = 135

[5,8] = 48

5 -> 8 = 40 + 8 = 48

8 -> 5 = 40 + 5 = 45

length = 3

[3,1,5] = 159

(1,3) -> 5 = 30 + 40 = 70

(1,5) -> 3 = 135 + 24 = 159

[1,5,8] =

(1,5) -> 8 = 135 + 24 =159

(5,8) -> 1 = 48 + 3

dp[i][j] =

nums[i-1]*nums[k]*nums[j+1]

+

dp[i][k-1]

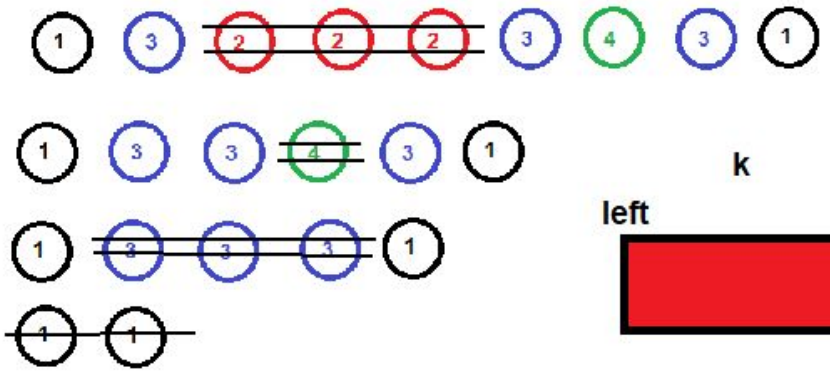
+

dp[k+1][j]

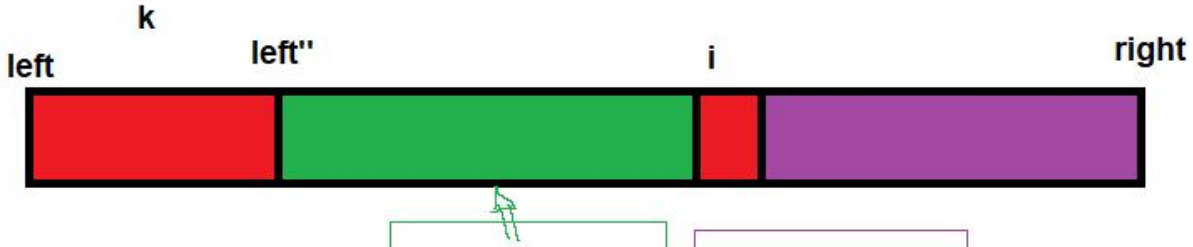
代码魔术师
magiciendecode.fr

546. Remove Boxes

https://leetcode.com/problems/remove-boxes/



dp[left][right][0]



result = Math.max(result, dfs(left + 1, i - 1, 0) + dfs(i, right, k + 1));

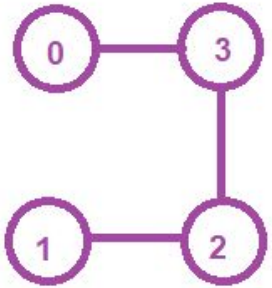
1. find consecutive box with same color, k++

2. for each box after, if the same color, try dfs(green part)+dfs(red+pink)

代码魔术师
magiciendecode.fr

547. Friend Circles

https://leetcode.com/problems/friend-circles/



f (unionFind)

```
private int unionFind(int[] union, int i) {
    return union[i] == i ? i : unionFind(union, union[i]);
}
```

[0 1 2 3]

i = 0, j = 3
unionFind(i) = 0
unionFind(j) = 3
array[0] = 3
[3 1 2 3]

[3 1 2 3]
i = 1, j = 2
unionFind(i) = 1
unionFind(j) = 2
array[1] = 2
[3 2 2 3]

[3 2 2 3]
i = 2, j = 3
unionFind(i) = 2
unionFind(j) = 3
array[2] = 3
[3 2 3 3]

[3 2 3 3] => unionFind(0) = unionFind(3) = 3
unionFind(1) = unionFind(2) = unionFind(3) = 3
unionFind(2) = unionFind(3) = 3
unionFind(3) = 3

代码魔术师
magiciendecode.fr

91. Decode Ways

https://leetcode.com/problems/decode-ways/

[2 2 6]

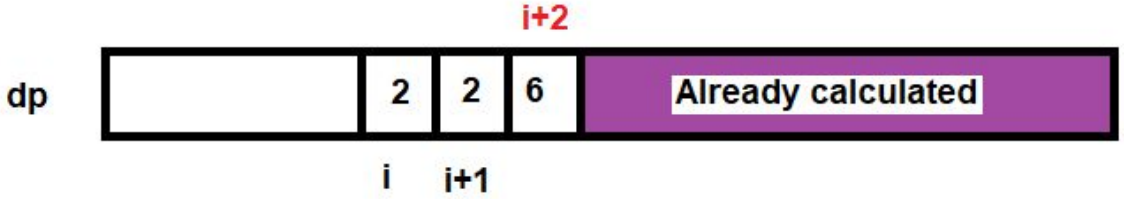
2 -> 26
22 -> 6
2 -> 2 -> 6

dfs(0), if dfs(length), result ++

Time Limit Exceeded

0 -->
03 -->
30 -->

i i+1
2 2 --> dfs(i+1) dfs(i+2)
3 0 --> dfs(i+1) --> KO
20 or 10 --> dfs(i+2)
9 3 --> dfs(i+1) -->



dfs(i) = dfs(i+1) + dfs(i+2)

dfs(i+1) = dfs(i+2) ...

```
if(dp != 0){
    return dp;
}
```

代码魔术师
magiciendecode.fr

472. Concatenated Words

https://leetcode.com/problems/concatenated-words/

a
a a
a a a
a a a a
a a a a a
a a a a a a
a a a a a a a
a a a a a a a a
a a a a a a a a b
b a a a a a a a a

PrefixTree

- PrefixTree[26]
- insert
- isEnd

1. sort the given string array based on length of word.

(if word1 can be built by word2 and word3, len(w1) > len(w2) and len(w1) > len(w3)

2. for each word, before verify it, do a check in contains array,
   if current contains can NOT cover the entire word, skip the verification.

3. add word to prefix tree, for each char in word, add it into contains array.

4. is valid or not ?

cat
cats
dog

verify(catsdog)

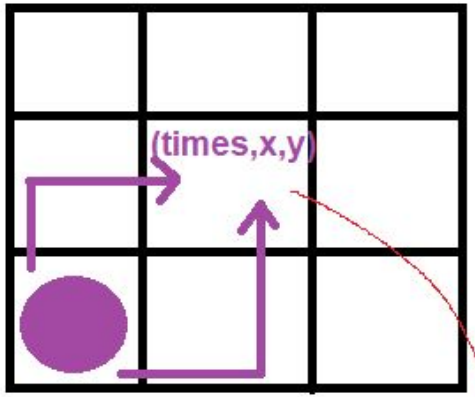
According to 1,2,3 we have all chars, and 3 words in prefix tree.
when iterate at red t, prefix tree isEnd, launch another bfs to verify (current index + 1). count++
So does the red s

if all chars can be iterated and count > 1, return true!

代码魔术师
magiciendecode.fr

576. Out of Boundary Paths

https://leetcode.com/problems/out-of-boundary-paths/



if the result of (times,x,y) has been calculated, just return it.

depth first seach ()

if not in bound, return 1
if time == 0, return 0

foreach direction, go with times -1

Time Limit Exceeded

Recursion with Memoization

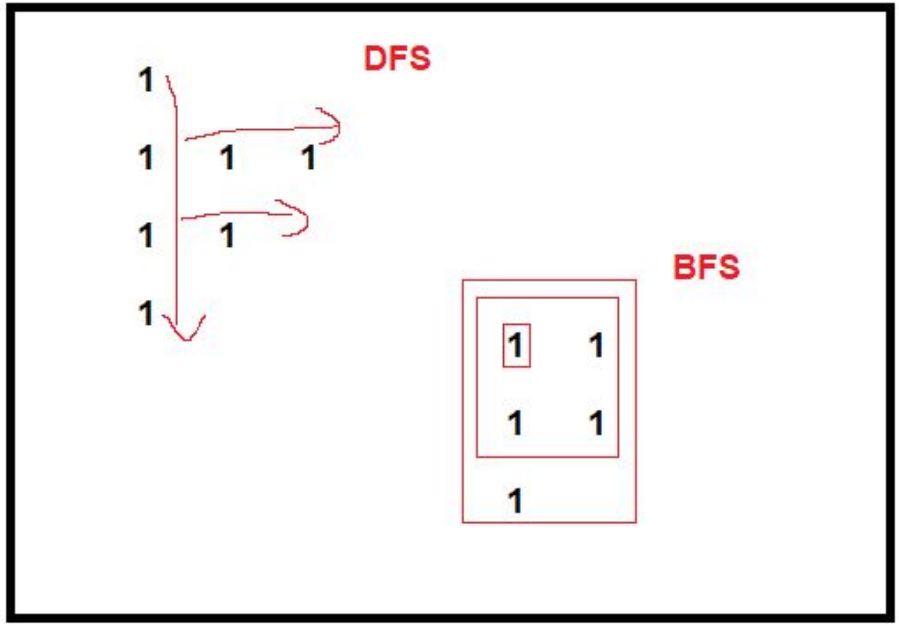
final int[][][] dp = m n times
default -1

if dp >= 0, return dp

代码魔术师
magiciendecode.fr

695. Max Area of Island

https://leetcode.com/problems/max-area-of-island/

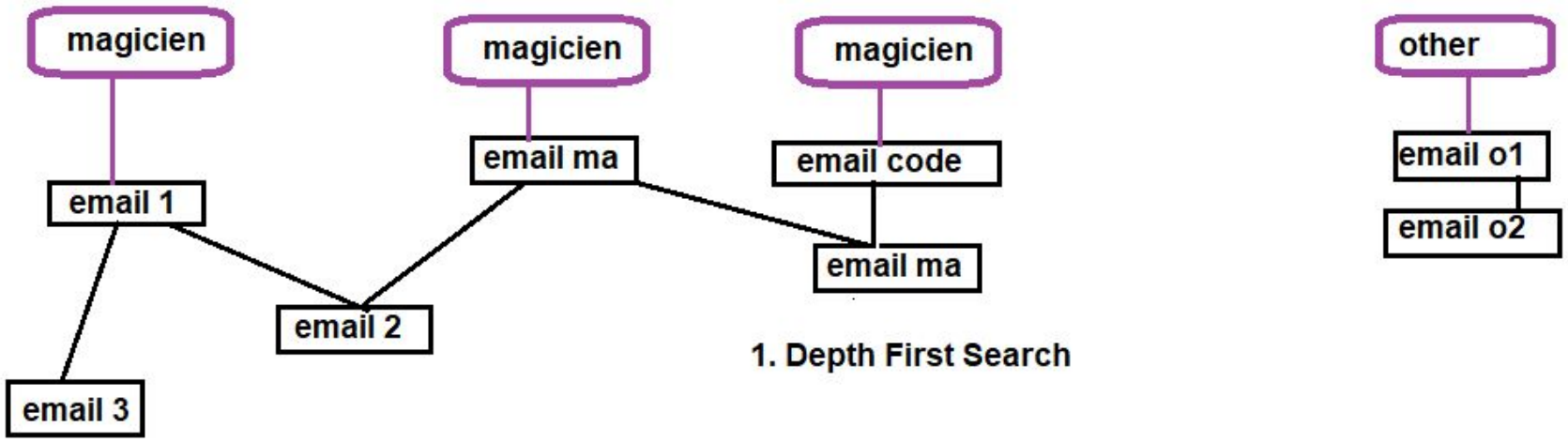


visited[][]

代码魔术师
magiciendecode.fr

721. Accounts Merge

https://leetcode.com/problems/accounts-merge/



1. Depth First Search

2. Union Find

代码魔术师
magiciendecode.fr

802. Find Eventual Safe States

https://leetcode.com/problems/find-eventual-safe-states/

foreach path
-- add node in a hashset,
-- if for a node, hashset contains its neighbor , return false

for a node, if at least one neighbor return false, then it's a unsafe node

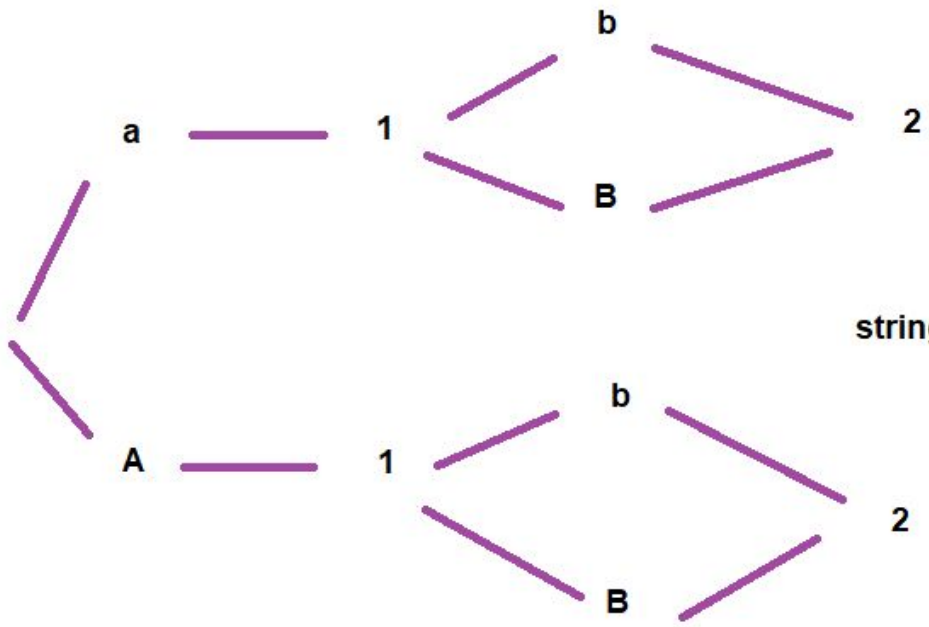
0 -> 1 -> 3 hashet(0,1,3)
then set.contains(1)
return false

0 -> 1 -> 2 -> 5

代码魔术师
magiciendecode.fr

784. Letter Case Permutation

https://leetcode.com/problems/letter-case-permutation/



dfs without for loop

stringBuilder.deleteCharAt(stringBuilder.length() - 1);

代码魔术师
magiciendecode.fr

645. Set Mismatch

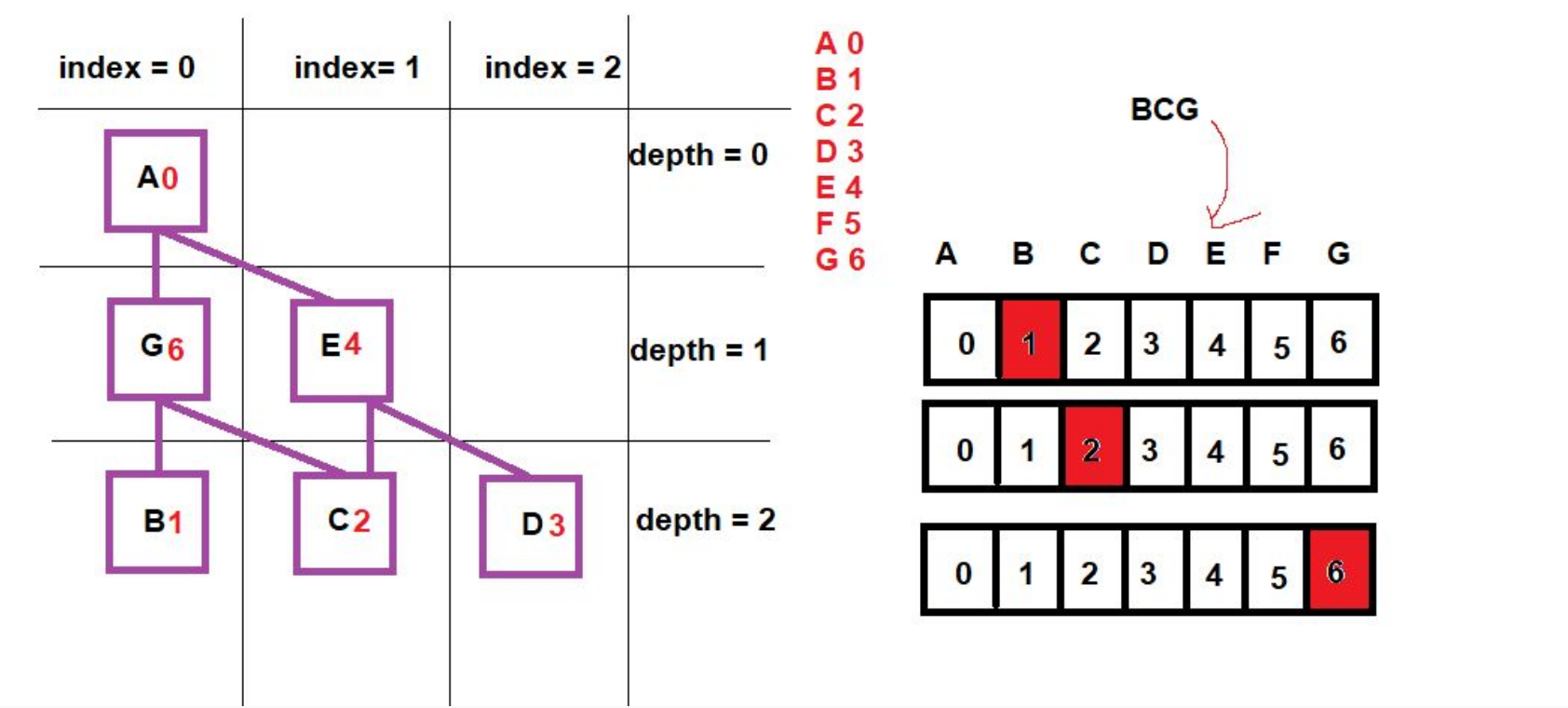
https://leetcode.com/problems/set-mismatch/

1. every slide should be reviewed
2. prepared at least one time

代码魔术师
magiciendecode.fr

756. Pyramid Transition Matrix

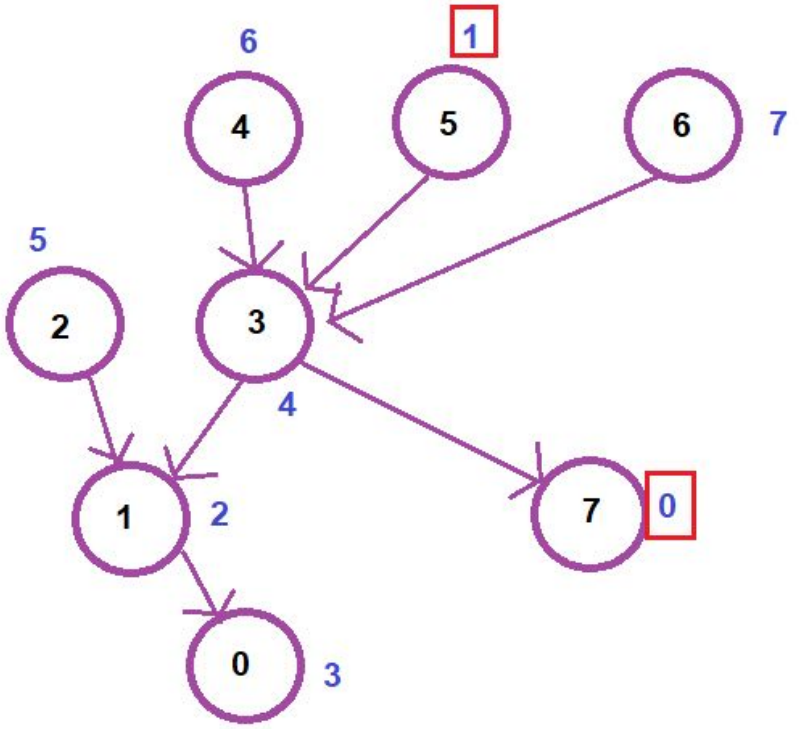
https://leetcode.com/problems/pyramid-transition-matrix/



代码魔术师
magiciendecode.fr

851. Loud and Rich

https://leetcode.com/problems/loud-and-rich/



1. build directional graph
   0 -> 1
   1 -> 2
   1 -> 3
   ...
2. for each index i
   dfs find the lowest queit from its neighbor and store the value in an array.

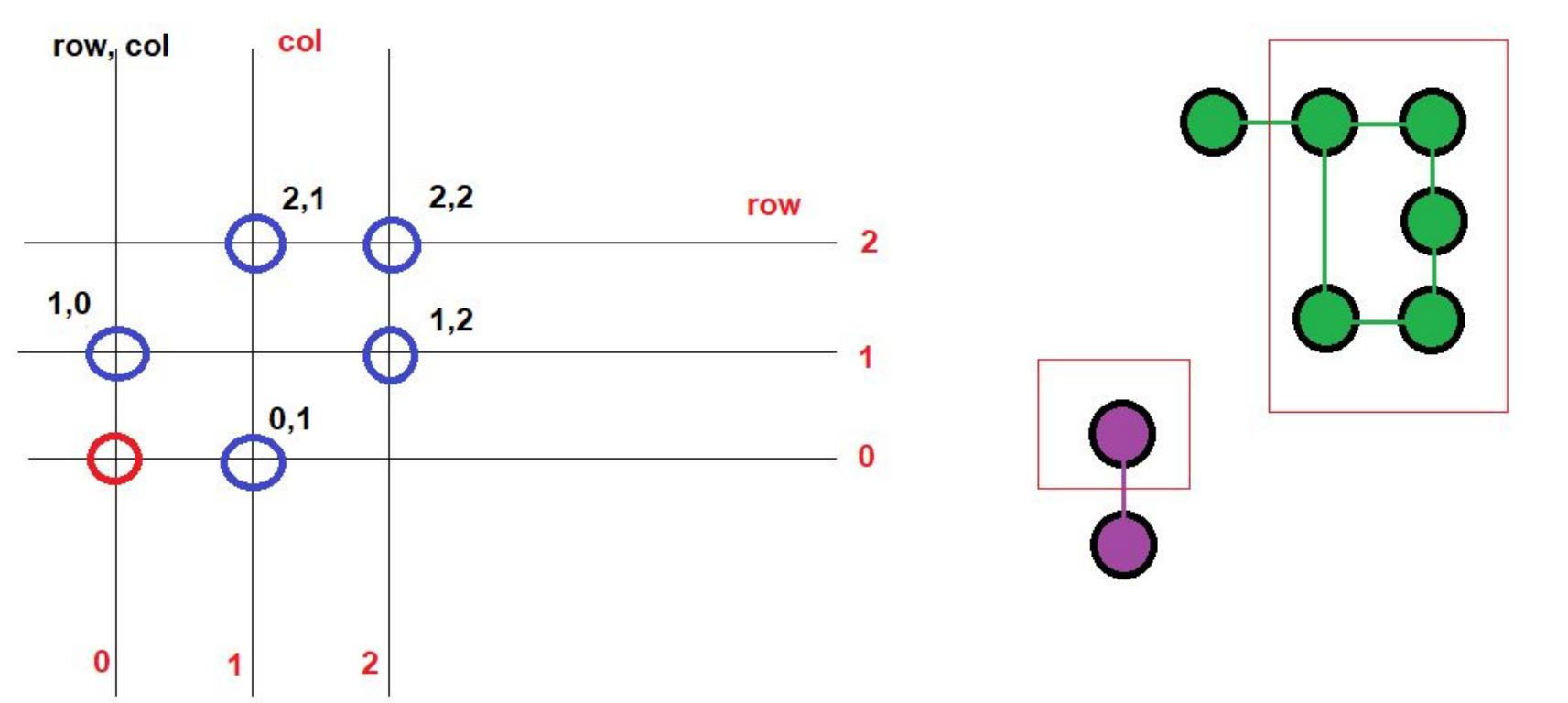
3. if for an index i that have had a value in array, return it directly.

1. 2. 3. are only main ideas,
more details are in the comments !

代码魔术师
magiciendecode.fr

947. Most Stones Removed with Same Row or Column

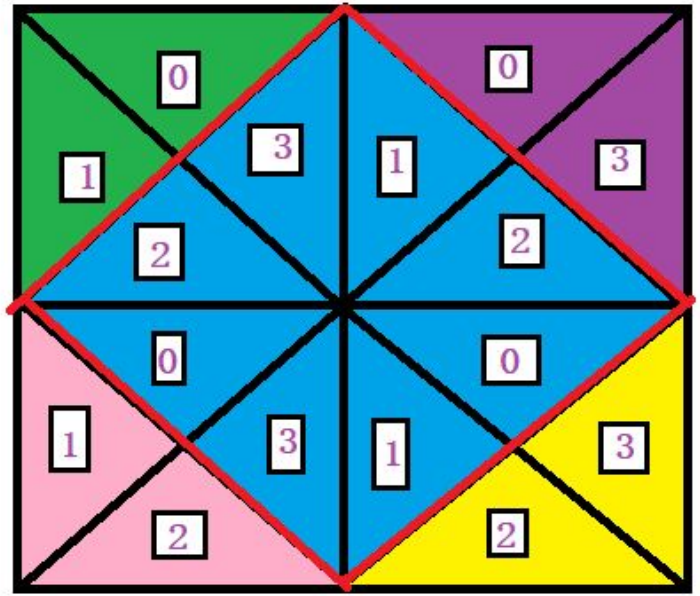
https://leetcode.com/problems/most-stones-removed-with-same-row-or-column/



代码魔术师
magiciendecode.fr

959. Regions Cut By Slashes

https://leetcode.com/problems/regions-cut-by-slashes/

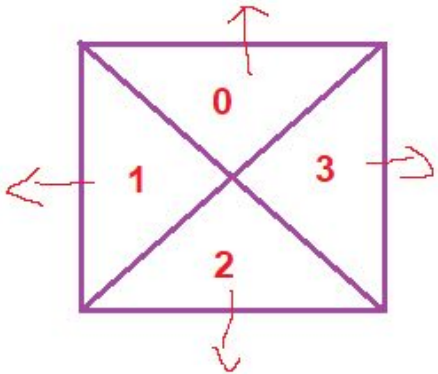


next row or col:

0	-> up	2
1	-> left	3
2	-> down	0
3	-> right	1

neighbors:

0 -> 1 3
1 -> 0 2
2 -> 1 3
3 -> 0 2

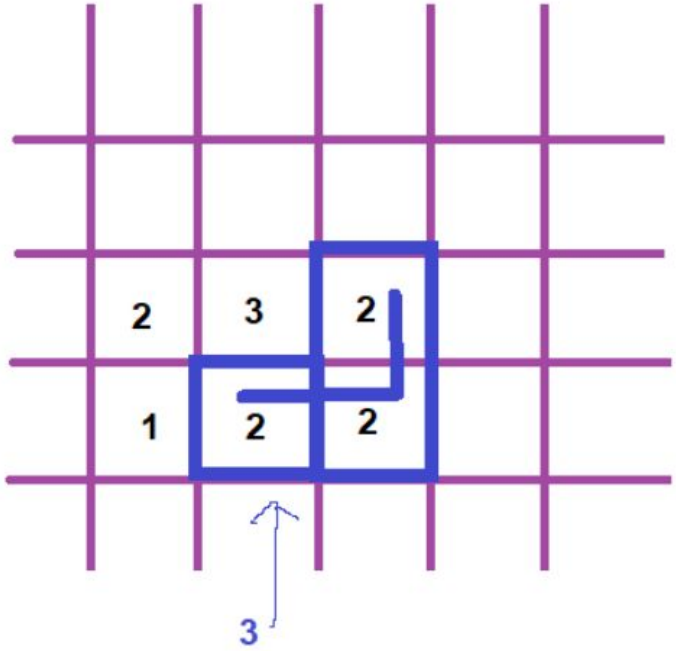


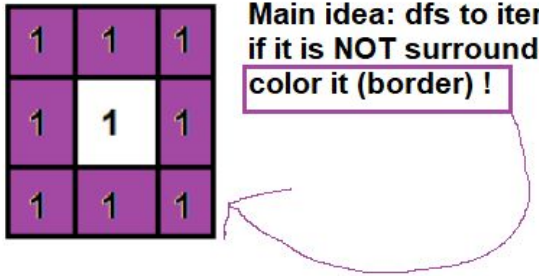
for neighbors, we should check whether they are connected or not.

代码魔术师
magiciendecode.fr

1034. Coloring A Border

https://leetcode.com/problems/coloring-a-border/





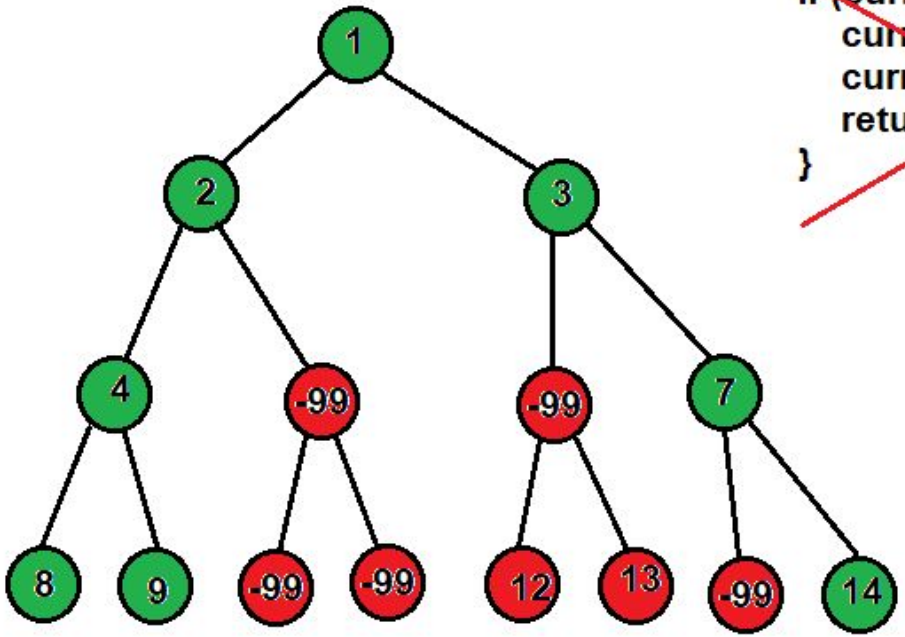
Main idea: dfs to iterate connected squares,
if it is NOT surrounded with 4 connected squares,
color it (border) !

The border of a connected component is all the squares in the connected component that are either 4-directionally adjacent to a square not in the component, or on the boundary of the grid (the first or last row or column).

代码魔术师
magiciendecode.fr

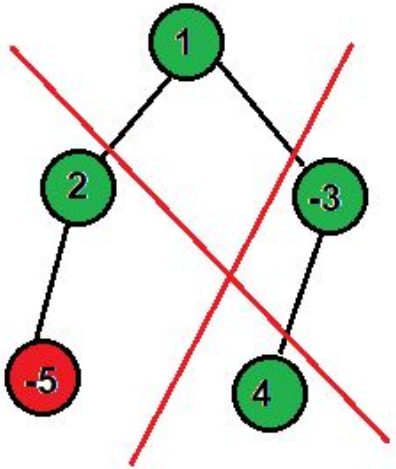
1080. Insufficient Nodes in Root to Leaf Paths

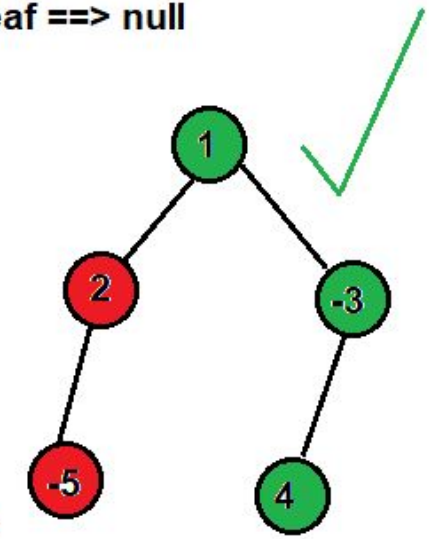
https://leetcode.com/problems/insufficient-nodes-in-root-to-leaf-paths/




```
if (current.left == null &&
    current.right == null &&
    currentSum < limit ) {
    return null;
}
```

```
if current node is a leaf:
    if sum < limit  ==> null
    else ==> current

if after dfs node becomes
a leaf ==> null
```





代码魔术师
magiciendecode.fr

1254. Number of Closed Islands

https://leetcode.com/problems/number-of-closed-islands/

1	1	1	1	1	1	1	0
1	0	0	0	0	1	1	0
1	0	1	0	1	1	1	0
1	0	0	0	0	1	0	1
1	1	1	1	1	1	1	0

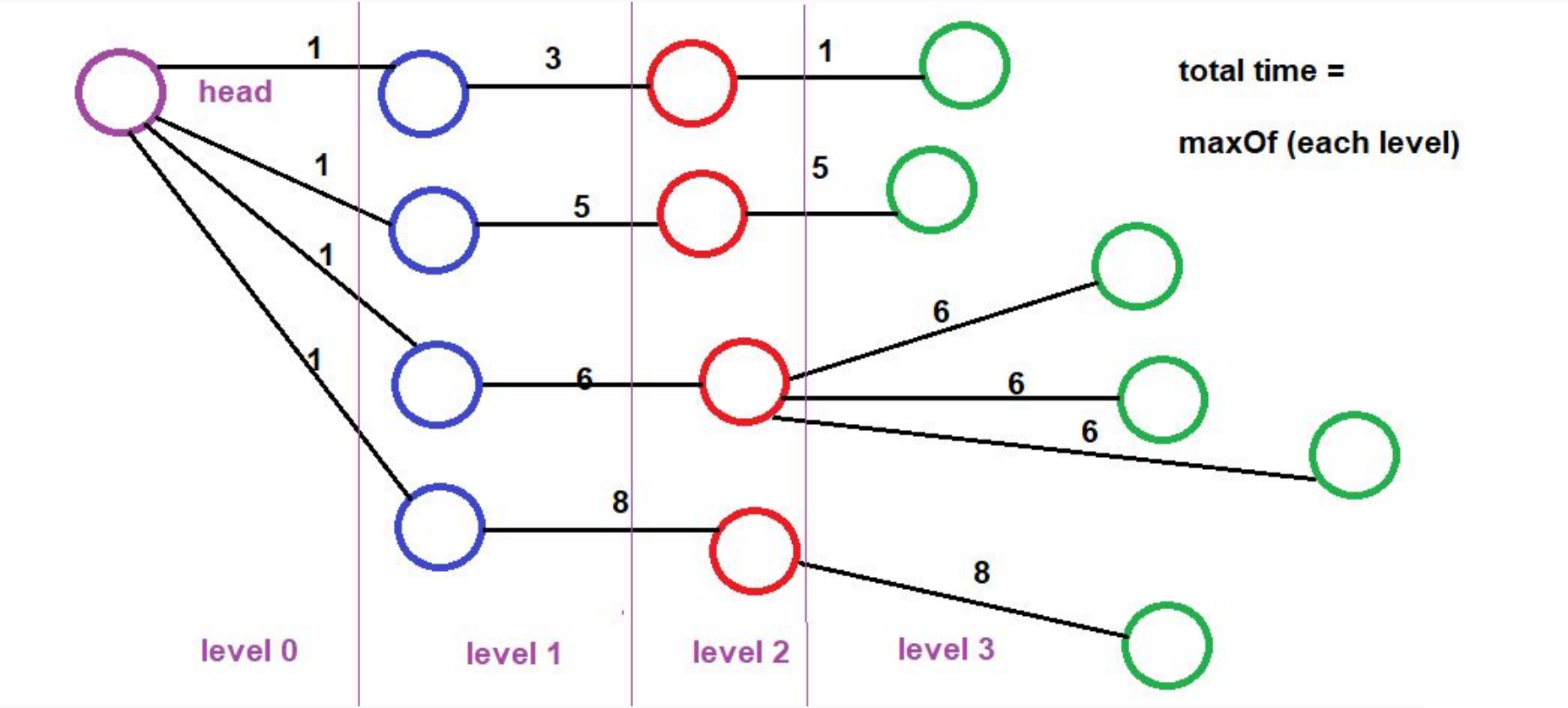
Key point:

closed islands:
all 4 directional neighbor squares of all connected squares 0 should in bound. (in grid)

Another idea is excluded all boundary connected squares 0 (mark as visited), then count the closed islands.

代码魔术师
magiciendecode.fr

1376. Time Needed to Inform All Employees



代码魔术师
magiciendecode.fr

1625. Lexicographically Smallest String After Applying Operations

https://leetcode.com/problems/lexicographically-smallest-string-after-applying-operations/

chars:
'0', '1', '2', '3', '4', '5', '6', '7', '8', '9'

from the starting string "xyzbla..."
compare with result, store min string
do apply a
do apply b
if (strA is not visited) dfs strA
if (strB is not visited) dfs strB

applyA:
for each odd index:
calculate next char,
= chars[(number of currentIndex +a)%10]
then set nextChar to currentIndex

apply B:
for each index:
calculate next index,
= (currentIndex+b)%length
then set currentChar to next index

代码魔术师
magiciendecode.fr

1722. Minimize Hamming Distance After Swap Operations

index	0	1	2
	1	3	2
	2	1	2

allowedSwaps= [[0,1],[0,2]]

out put = 1

if two index in allowedSwaps, that means they are in the same group, so unionFind(i) will return the same value.

for each unionFind returned value, create a Map<Int,Int> to recored the number and count.

Map<Integer,Map<Integer,Integer>>

count

unionFind return value

number

foreach index in target array,
calculate unionFind value, then get related map
check count, if there are enough number
if count > 0, ++same

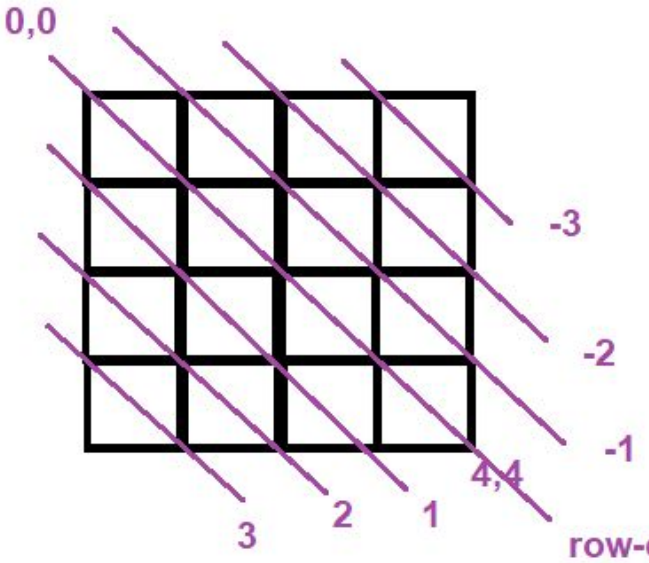
target.length - same

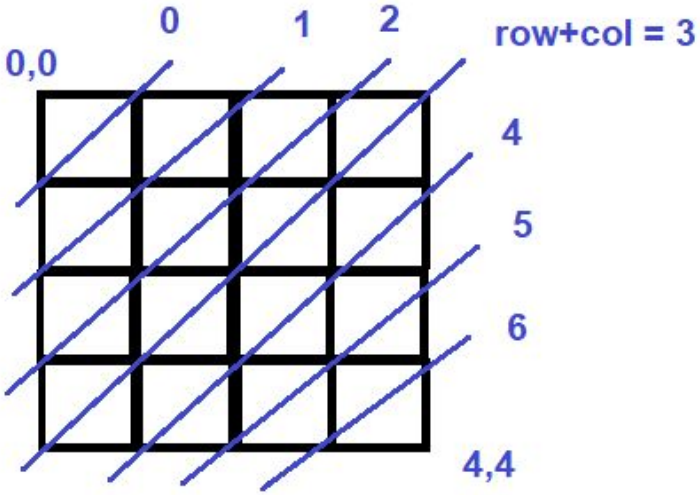
代码魔术师
magiciendecode.fr

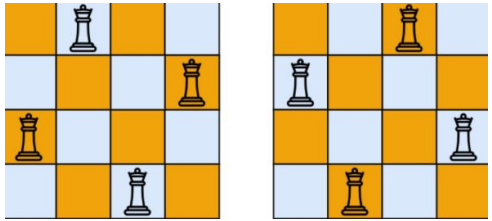
51. N-Queens

https://leetcode.com/problems/n-queens/

```
iterate by row
for col in [0,n) :

if  visited one of
--> diagonal 1
--> diagonal 2
--> col
            continue;

add Queen

dfs ( row + 1 )

remove Queen
```



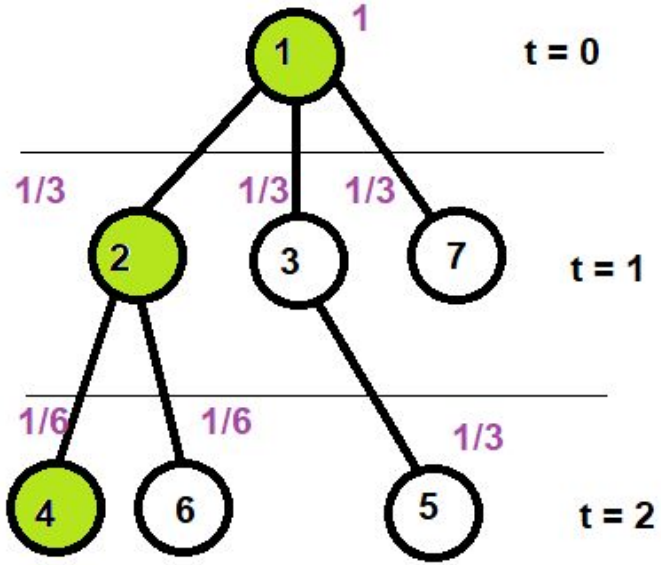




代码魔术师
magiciendecode.fr

1377. Frog Position After T Seconds

https://leetcode.com/problems/frog-position-after-t-seconds/



1. build graph

2. initialize array times and probabilities

3. dfs from node 1 with probability 1

DFS:
if currentTime > t (level > t), return

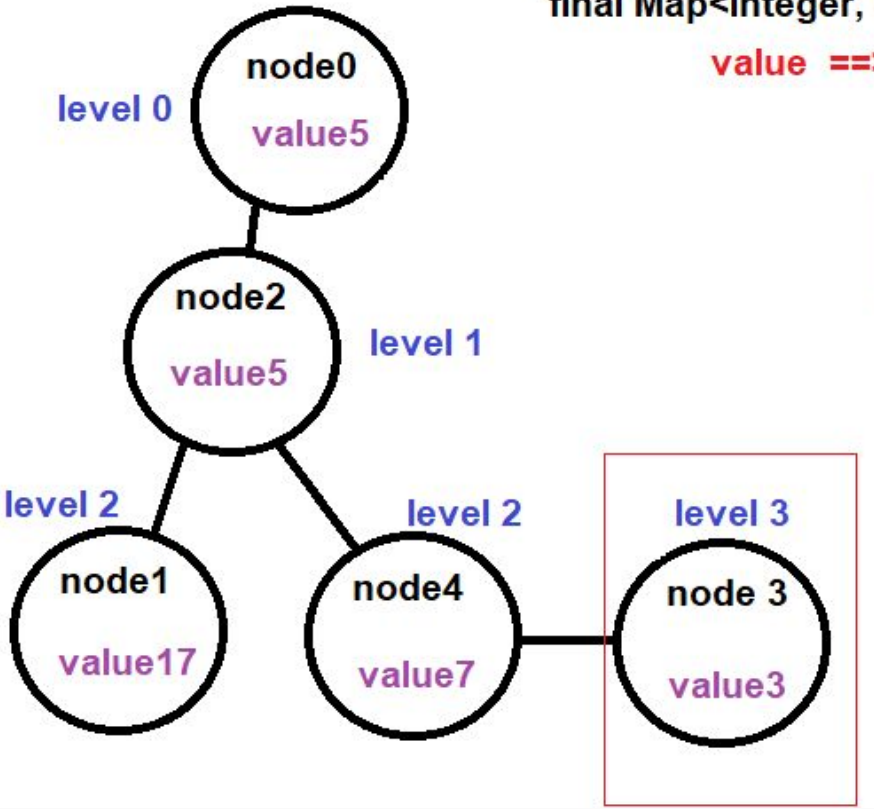
count unvisited neighbors that have probability 0.

for each neighbor
    assign probability,
    then dfs next level.

代码魔术师
magiciendecode.fr

1766. Tree of Coprimes

https://leetcode.com/problems/tree-of-coprimes/




```
final Map<Integer, Stack<int[]>> valueAncestors = new HashMap<>();
```

value ==> [node,level]

key

value5	stack([node0,level0], [node2,level1])
value7	stack([node4,level2])

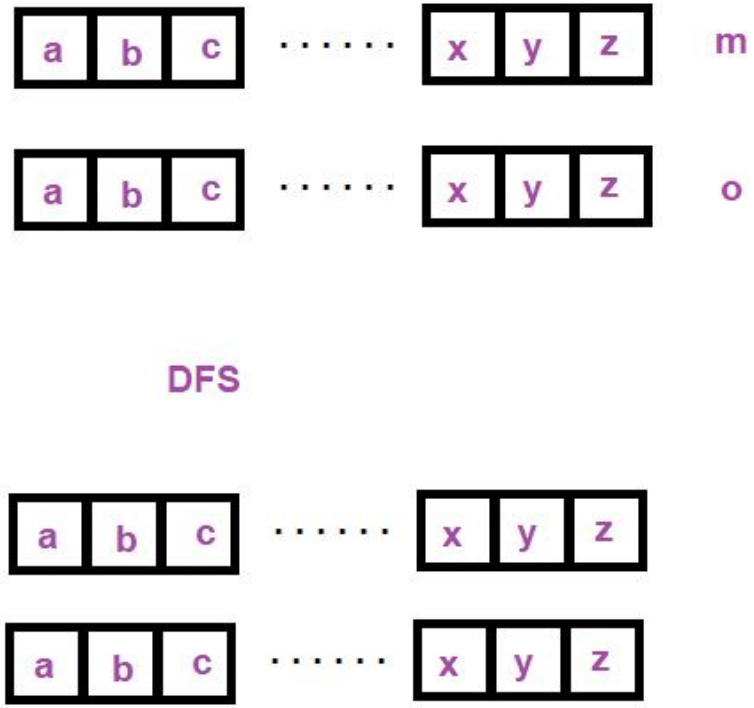
For value3, calculate gcd with each key in map:
then take the top element and compare level:

the closest ancestor == max(level)

代码魔术师
magiciendecode.fr

1268. Search Suggestions System

https://leetcode.com/problems/search-suggestions-system/



1. add words to tree

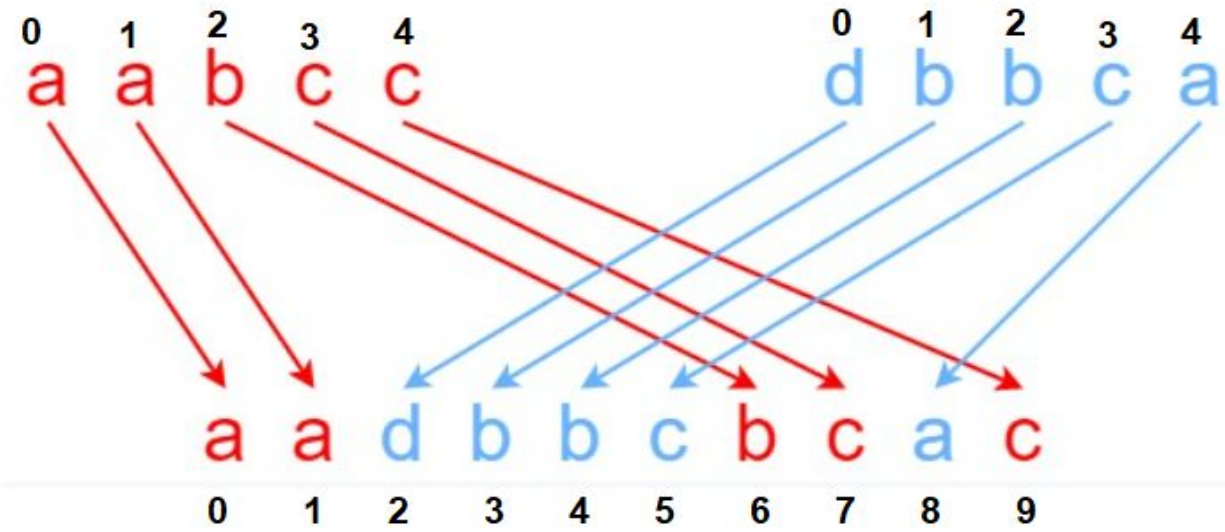
2. for each prefix, do dfs

(a) search the level that has exactly prefix
(b) if cannot find prefix, add empty list
(c) dfs to take the first 3 results

代码魔术师
magiciendecode.fr

97. Interleaving String

https://leetcode.com/problems/interleaving-string/

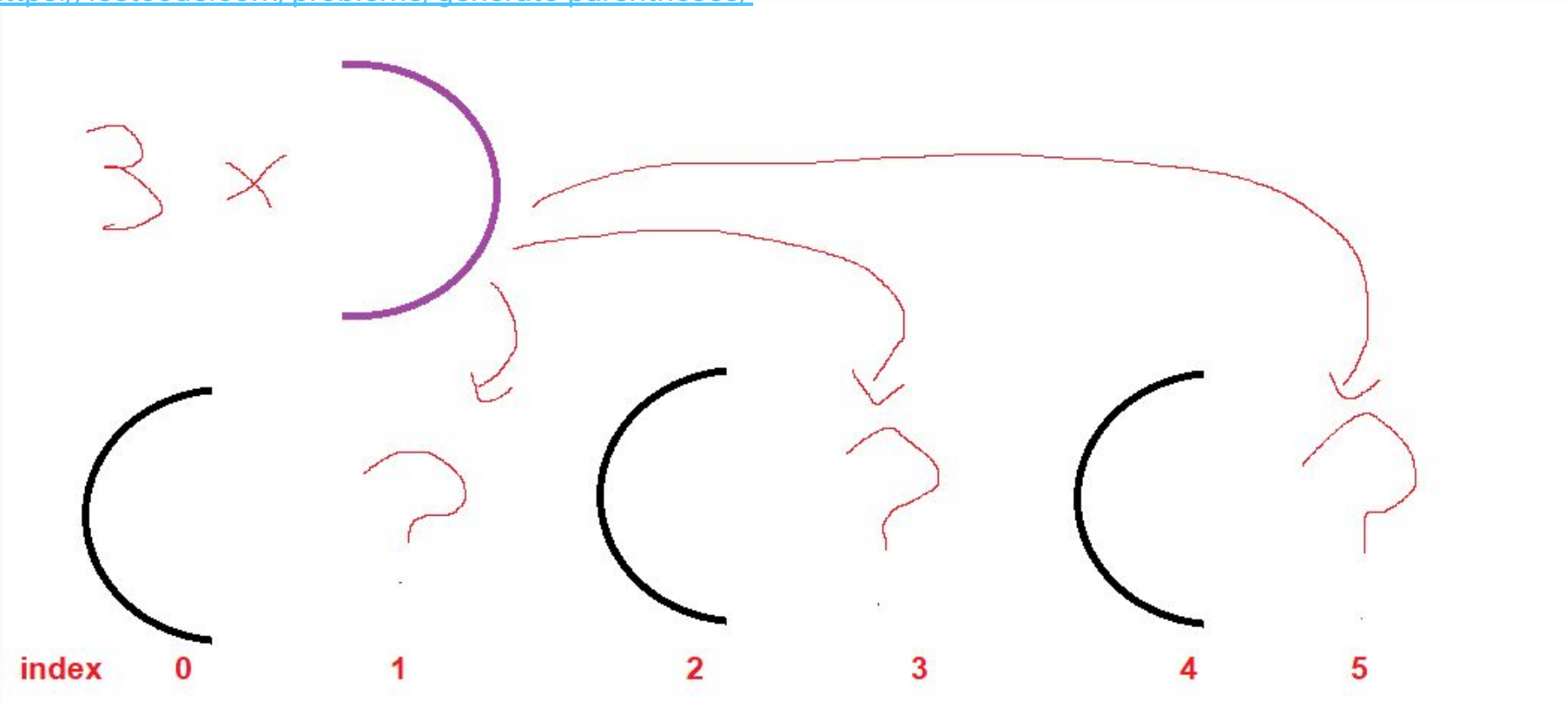


s1 index == s1.length
&&
s2 index == s2.length
&&
target index == target.length

代码魔术师
magiciendecode.fr

22. Generate Parentheses

https://leetcode.com/problems/generate-parentheses/



代码魔术师
magiciendecode.fr