60. Permutation Sequence

給出數字n，你可以使用數字1,2，。。。n。將所有排列成的數從小到大排成一列，找出排在第K的數

思路：在arraylist中放入1到n。factorial[]數組，factorial [ I ]表示i的階乘。k=k-1

例如n=4, 若要找出第一位數，就將index=K/factorial[4-1] 然後第一位數就是arraylist(index),加入string后，記住在list中刪除這個數字，k=k%factorial [4-1]

public String getPermutation(int n, int k) {

List<Integer> num = new LinkedList<Integer>();

for (int i = 1; i <= n; i++) num.add(i);

int[] fact = new int[n]; // factorial

fact[0] = 1;

for (int i = 1; i < n; i++) fact[i] = i\*fact[i-1];

k = k-1;

StringBuilder sb = new StringBuilder();

for (int i = n; i > 0; i--){

int ind = k/fact[i-1];

k = k%fact[i-1];

sb.append(num.get(ind));

num.remove(ind);

}

return sb.toString();

}

### 756. Pyramid Transition Matrix

[My Submissions](https://leetcode.com/contest/weekly-contest-65/problems/pyramid-transition-matrix/submissions/)[Back to Contest](https://leetcode.com/contest/weekly-contest-65/)

* **User Accepted:**58
* **User Tried:**219
* **Total Accepted:**58
* **Total Submissions:**534
* **Difficulty:Medium**

We are stacking blocks to form a pyramid. Each block has a color which is a one letter string, like `'Z'`.

For every block of color `C` we place not in the bottom row, we are placing it on top of a left block of color `A` and right block of color `B`. We are allowed to place the block there only if `(A, B, C)` is an allowed triple.

We start with a bottom row of bottom, represented as a single string. We also start with a list of allowed triples allowed. Each allowed triple is represented as a string of length 3.

Return true if we can build the pyramid all the way to the top, otherwise false.

**Example 1:**

**Input:** bottom = "XYZ", allowed = ["XYD", "YZE", "DEA", "FFF"]**Output:** true**Explanation:**

We can stack the pyramid like this:

A

/ \

D E

/ \ / \

X Y Z

This works because ('X', 'Y', 'D'), ('Y', 'Z', 'E'), and ('D', 'E', 'A') are allowed triples.

**Example 1:**

**Input:** bottom = "XXYX", allowed = ["XXX", "XXY", "XYX", "XYY", "YXZ"]**Output:** false**Explanation:**

We can't stack the pyramid to the top.

Note that there could be allowed triples (A, B, C) and (A, B, D) with C != D.

**Note:**

1. bottom will be a string with length in range [2, 12].
2. allowed will have length in range [0, 343].
3. Letters in all strings will be chosen from the set {'A', 'B', 'C', 'D', 'E', 'F', 'G'}.

题目： 给出一个金字塔底座，和一个可以允许的三角列表[ abc, bcd]

Abc 代表 允许左下a， 右下b 上c 的三角形组合存在。

问这个列表和这个底座，能不能堆出一座金字塔 堆到塔尖

思路：dfs 或bfs 加记忆化

我们可以对每一层进行bfs

每一层数据结构如下

[ [这层的这个位置可以放进去的所有可能性] [] [] [] ]

用例子说明 xyz allowed: xyx xya yza

第一层[ [x] [y] [z] ]

第二层[ [x ,a] [a] ]

第三层[ [xa 和 aa 两个组合都没法往这个set里塞东西] ] fail fail条件是某层某个位置算完后，一个可能性都没有 自然无法继续往上堆。就可以断定最后答案

**public boolean** pyramidTransition(String bottom, List<String> allowed) {  
 LinkedList<Set<Character>> curr = **new** LinkedList<>();  
 **for**(**int** i = 0; i < bottom.length(); i ++){  
 curr.add(**new** HashSet<Character>());  
 curr.get(i).add(bottom.charAt(i));  
 }  
  
 HashMap<String, Set<Character>> memo = **new** HashMap<>();  
 **for**(String allow : allowed){  
 String key = allow.substring(0, 2);  
 **if**(!memo.containsKey(key)) memo.put(key, **new** HashSet<Character>());  
 memo.get(key).add(allow.charAt(2));  
 }  
  
 StringBuilder sb = **new** StringBuilder();  
 **while**(curr.size() > 1){  
 LinkedList<Set<Character>> next = **new** LinkedList<>();  
 **for**(**int** i = 0; i < curr.size() - 1; i ++){  
 HashSet<Character> set = **new** HashSet<Character>();  
  
 **for**(**char** left : curr.get(i)){  
 **for**(**char** right : curr.get(i + 1)){  
 sb.append(left).append(right);  
 **if**(memo.containsKey(sb.toString())) set.addAll(memo.get(sb.toString()));  
 sb.setLength(0);  
 }  
 }  
  
 **if**(set.size() == 0)  
 **return false**;  
  
 next.add(set);  
 }  
  
 curr = next;  
 }  
  
 **return true**;  
}