95. Unique Binary Search Trees II

题目描述：https://leetcode.com/problems/unique-binary-search-trees-ii/

> 求给定1-n的节点所能组成的二叉平衡树。

解题思路：

和Unique Binary Search Tree I 思路相同，但是f[i][j]是一个vector，存储着由i->j的节点所能组成的所有树

代码：

```
/**
 * Definition for a binary tree node.
 * struct TreeNode {
 *     int val;
 *     TreeNode *left;
 *     TreeNode *right;
 *     TreeNode(int x) : val(x), left(NULL), right(NULL) {}
 * };
 */
class Solution {
public:
    vector<TreeNode*> generateTrees(int n) {
        if(n == 0)
            return {};
        vector<vector<vector<TreeNode*> > > f(n+1, vector<vector<TreeNode*>>(n+1))
;
        for(int i = 1; i <= n; i++) {
            f[i][i] = {new TreeNode(i)};
            f[i][i][0]->left = NULL;
            f[i][i][0]->right = NULL;
        }
        for(int l = 2; l <= n; l++) {
            for(int i = 1; i + l - 1 <= n; i++) {
                int j = i + l - 1;
                for(auto item: f[i][j-1]) {
                    TreeNode *p = new TreeNode(j);
                    p->left = item;
                    p->right = NULL;
                    f[i][j].push_back(p);
                }
                for(auto item: f[i+1][j]) {
                    TreeNode *p = new TreeNode(i);
```

```
                    p->left = NULL;
                    p->right = item;
                    f[i][j].push_back(p);
                }
                for(int k = i + 1; k < j; k++) {
                    for(auto iteml: f[i][k-1]) {
                        for(auto itemr: f[k+1][j]) {
                            TreeNode *p = new TreeNode(k);
                            p->left = iteml;
                            p->right = itemr;
                            f[i][j].push_back(p);
                        }
                    }
                }

            }
        }
        return f[1][n];
    }
};
```