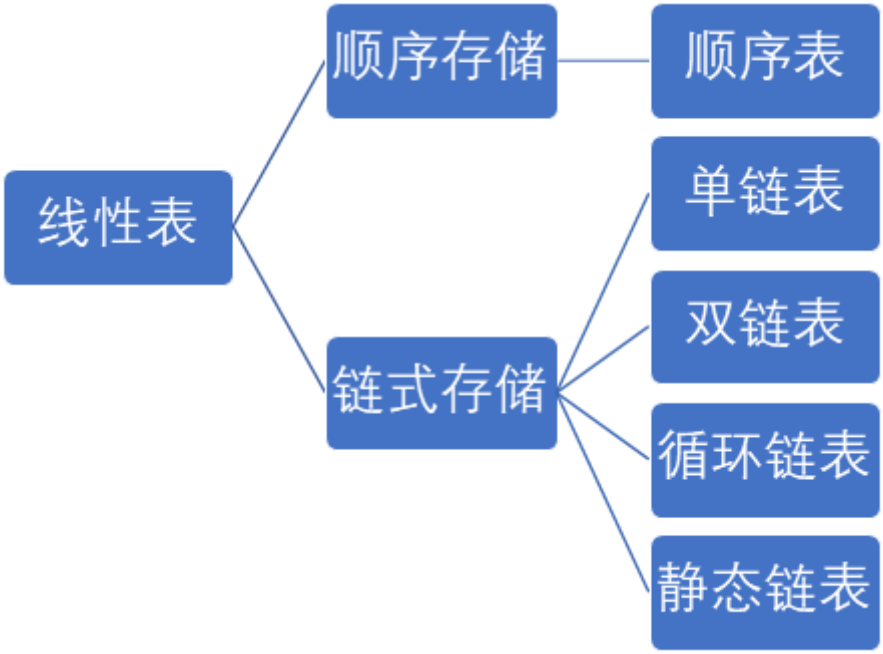
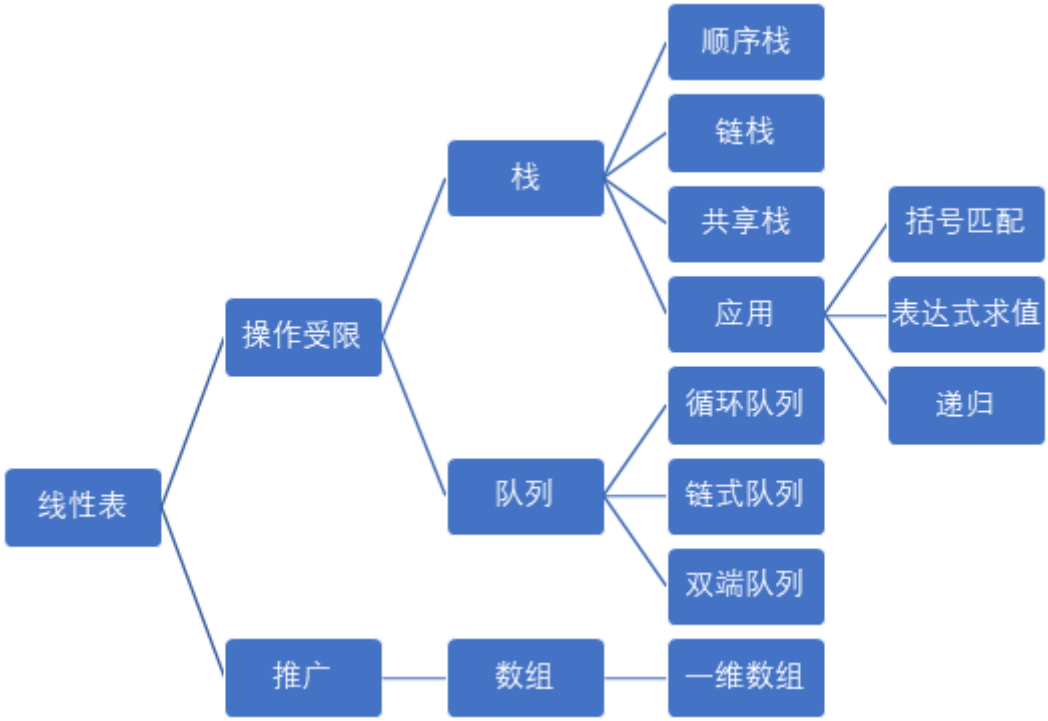
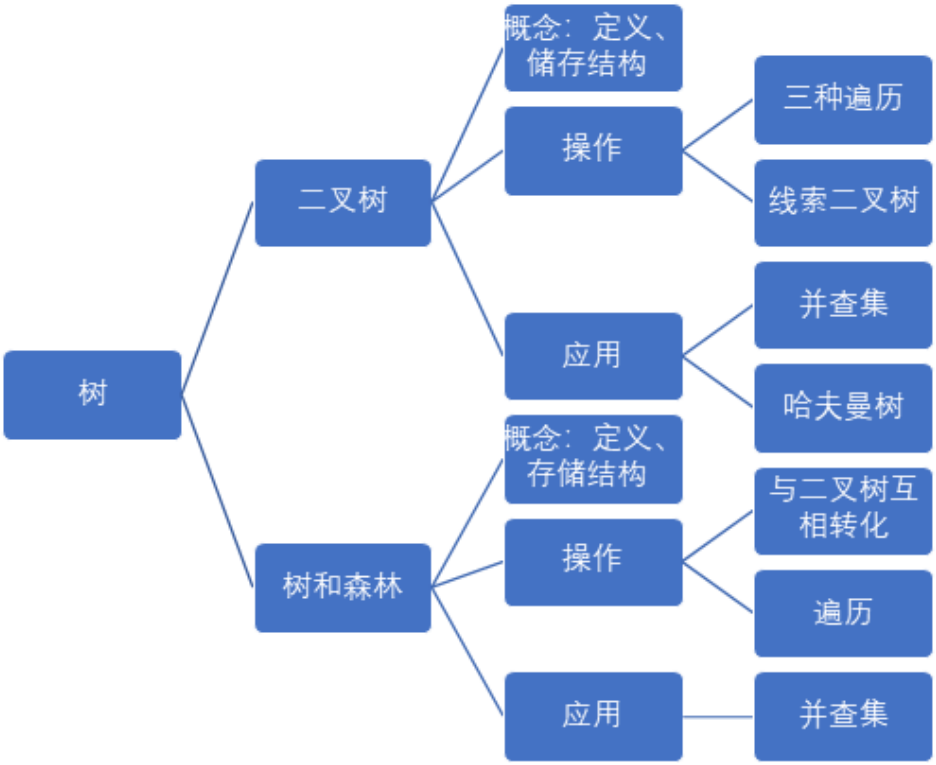
1. 知识图谱

1.1 线性表

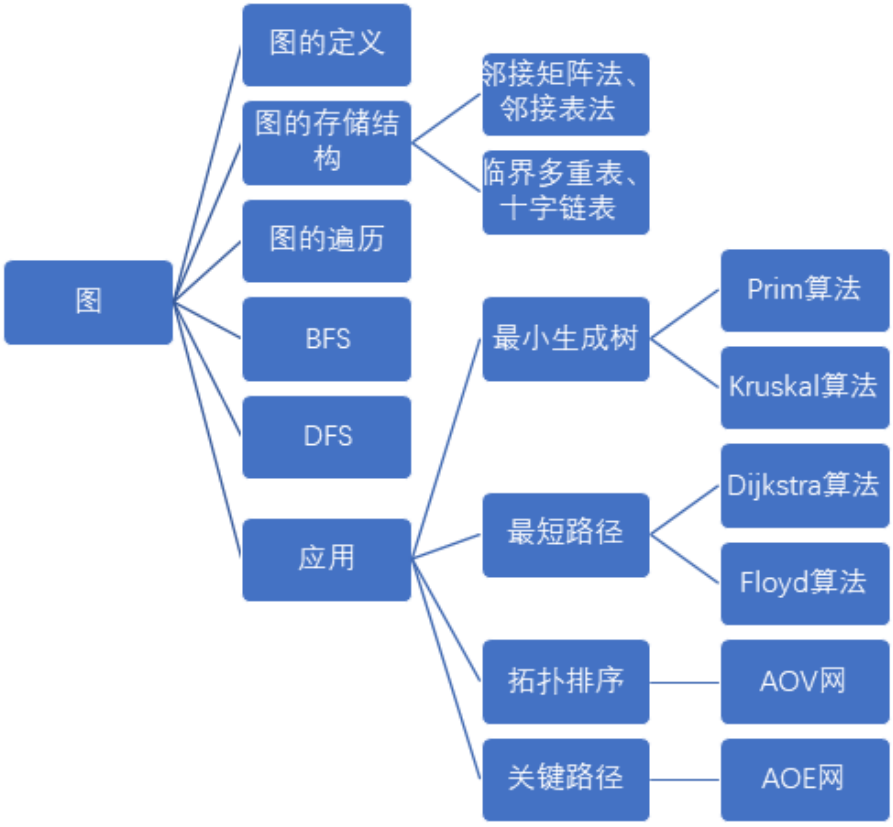




1.2 树



1.3 图



2. 每章习题

2.1 线性表

1. 题目 **1**

```
for (i = 1, s = 0; i <= n; i++) {
    t = 1;
    for (j = 1; j <= i; j++)
        t = t * j;
    s = s + t;
}
```

该程序的时间复杂度为?

答案: $O(n^2)$

---

2. 题目 **2**

对用邻接矩阵表示的具有n个顶点和e条边的图进行任一种遍历时，其时间复杂度为?
对用邻接表表示的图进行任一种遍历时，其时间复杂度为?

答案: $O(n^2), O(e)$

---

3. 题目 **3**

设某顺序表中第一个元素的地址是se（下标从1开始），每个结点占m个单元，则第i个结点的地址为:
A. $se - i \times m$
B. $se + (i - 1) \times m$
C. $se + i \times m$
D. $se + (i + 1) \times m$

答案: B

2.2 栈和队列

4. 题目 **4**

若用一个大小为6的数值来实现循环队列，且当前 `rear` 和 `front` 的值分别为 0 和 3，当从队列中删除一个元素，再加入两个元素后，`rear` 和 `front` 的值分别为?

答案: 2 和 4

---

5. 题目 **5**

若栈采用顺序存储方式存储，现两栈共享空间 `v[1…m]`，`top[1]`、`top[2]` 分别代表第1和第2个栈的栈顶，栈1的底在 `v[1]`，栈2的底在 `v[m]`，则栈满的条件是:
A. $|top[2] - top[1]| = 0$
B. $top[1] + 1 = top[2]$
C. $top[1] + top[2] = m$
D. $top[1] = top[2]$

答案: B

---

6. 题目 **6**

用不带头结点的单链表存储队列，其头指针指向队头结点，尾指针指向队尾结点，则在进行出队操作时:
A. 仅修改队头指针
B. 仅修改队尾指针

C．队头、队尾指针都可能要修改
D．队头、队尾指针都要修改

答案:C

2.3 串、数组和广义表

7. 题目 **7**

假设以行序为主序存储二维数组 $A=\text{array}[1\ldots100,1\ldots100]$，设每个数据元素占 2 个存储单元，基地址为 10，则 $LOC[5,5]=$ ？

答案: 818

---

8. 题目 **8**

设二维数组 $A[1\ldots m,1\ldots n]$ 按行存储在数组 B 中，则二维数组元素 $A[i,j]$ 在一维数组 B 中的下标为？

答案: $n\cdot i-n+j$

---

9. 题目 **9**

已知广义表 $L=((a,b,c),(d,e,f))$，运用下列( )运算可以得到结果 e：
A. $head(tail(L))$
B. $tail(head(L))$
C. $head(tail(head(tail(L))))$
D. $head(tail(tail(head(L))))$

答案: C

2.4 树和二叉树

10. 题目 **10**

利用n个权值作为叶子结点生成的哈夫曼树中共包含多少个结点？

答案: : $2n-1$

---

11. 题目 **11**

设后序遍历某二叉树的序列为 `DABEC`，中序遍历该二叉树的序列为 `DEBAC`，则前序遍历该二叉树的序列为？

答案: CEDBA

---

12. 题目 **12**

一个带权无向连通图的最小生成树（ ）：
A. 有一棵或多棵
B. 只有一棵
C. 一定有多棵
D. 不知道

答案: A

2.5 图

13. 题目 **13**

在含n个顶点和e条边的无向图的邻接矩阵中，零元素的个数为多少？

答案: $n^2 - 2e$

---

14. 题目 **14**

一个有向图中，某个顶点 V_i 的出度为 4，入度为 3，那么这个图的邻接表中，第i个链表的长度为？

答案: 4

---

15. 题目 **15**

图的 BFS 生成树的树高比 DFS 生成树的树高 ( )：
A．小
B．相等
C．小或相等
D．大或相等

答案: C

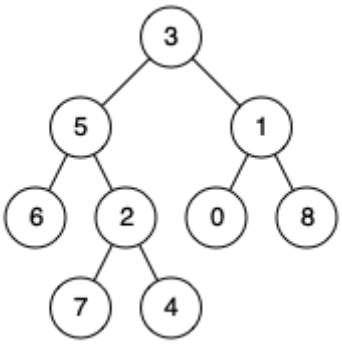
3. 挑战题

3.1 二叉树的最近公共祖先

给定一个二叉树, 找到该树中两个指定节点的最近公共祖先。

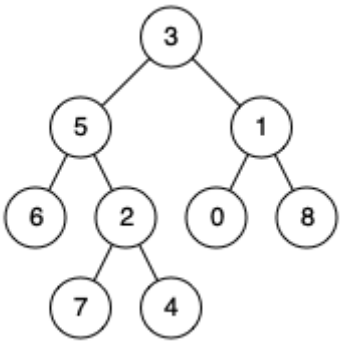
百度百科中最近公共祖先的定义为：“对于有根树 T 的两个节点 p、q，最近公共祖先表示为一个节点 x，满足 x 是 p、q 的祖先且 x 的深度尽可能大（一个节点也可以是它自己的祖先）。”

示例 1：




```
输入：root = [3,5,1,6,2,0,8,null,null,7,4], p = 5, q = 1
输出：3
解释：节点 5 和节点 1 的最近公共祖先是节点 3 。
```

示例 **2**：




```
输入：root = [3,5,1,6,2,0,8,null,null,7,4], p = 5, q = 4
输出：5
解释：节点 5 和节点 4 的最近公共祖先是节点 5 。因为根据定义最近公共祖先节点可以为节点本身。
```

示例 **3**：

```
输入：root = [1,2], p = 1, q = 2
输出：1
```

提示：

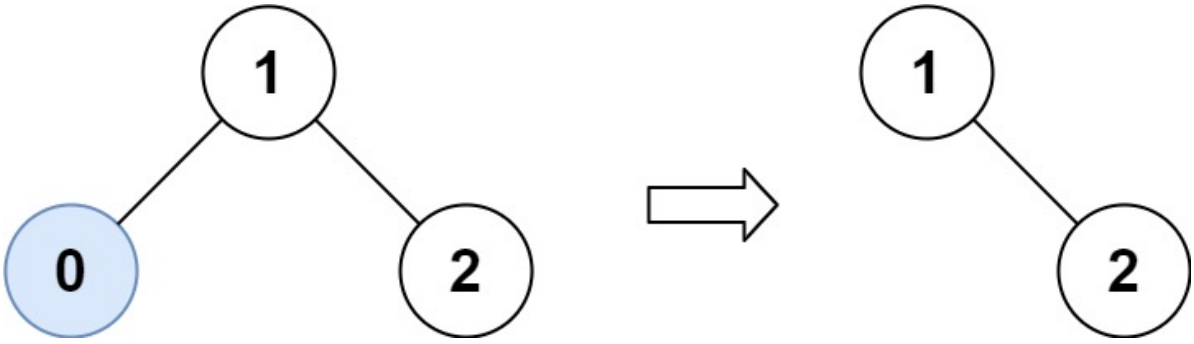
- 树中节点数目在范围 `[2, 105]` 内。
- `-109 <= Node.val <= 109`
- 所有 `Node.val` `互不相同` 。
- `p != q`
- `p` 和 `q` 均存在于给定的二叉树中。

3.2 修剪二叉搜索树

给你二叉搜索树的根节点 `root` ，同时给定最小边界`low` 和最大边界 `high`。通过修剪二叉搜索树，使得所有节点的值在`[low, high]`中。修剪树 **不应该** 改变保留在树中的元素的相对结构 (即，如果没有被移除，原有的父代子代关系都应当保留)。 可以证明，存在 **唯一的答案** 。

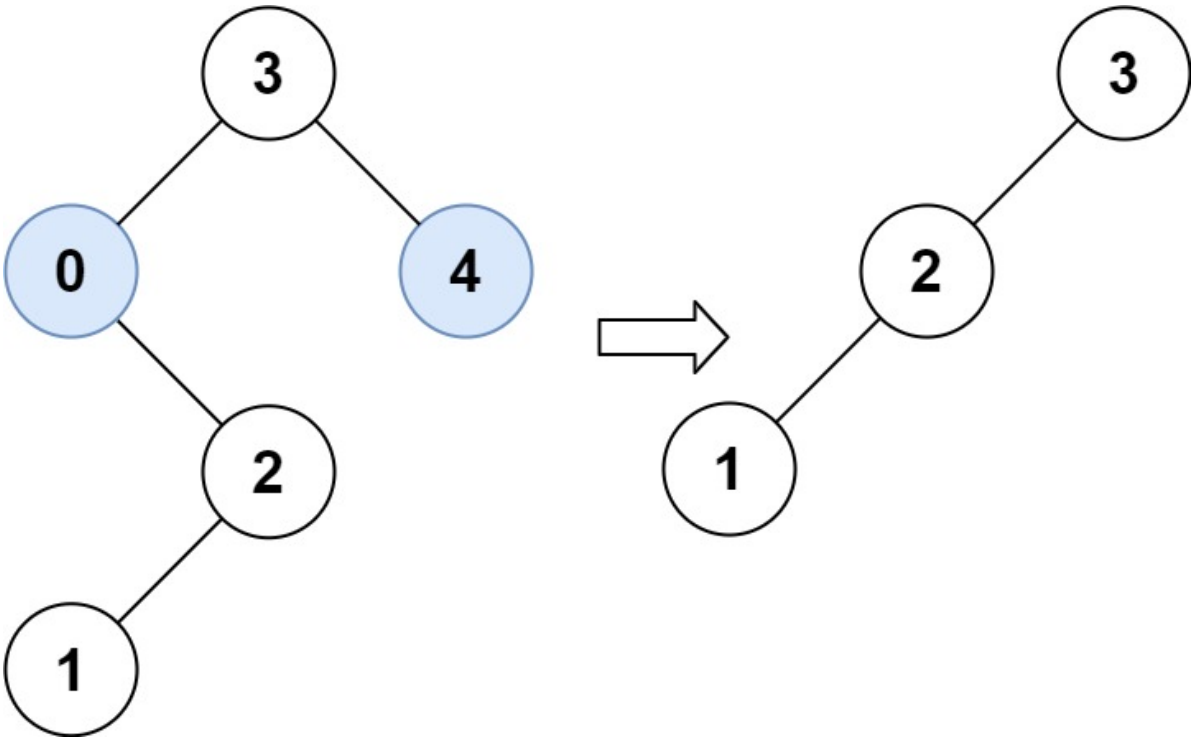
所以结果应当返回修剪好的二叉搜索树的新的根节点。注意，根节点可能会根据给定的边界发生改变。

示例 **1**：




```
输入：root = [1,0,2], low = 1, high = 2
输出：[1,null,2]
```

示例 **2**：




```
输入：root = [3,0,4,null,2,null,null,1], low = 1, high = 3
输出：[3,2,null,1]
```

提示：

- 树中节点数在范围 `[1, 104]` 内
- `0 <= Node.val <= 104`
- 树中每个节点的值都是 **唯一** 的
- 题目数据保证输入是一棵有效的二叉搜索树
- `0 <= low <= high <= 104`

3.3 太平洋大西洋水流问题

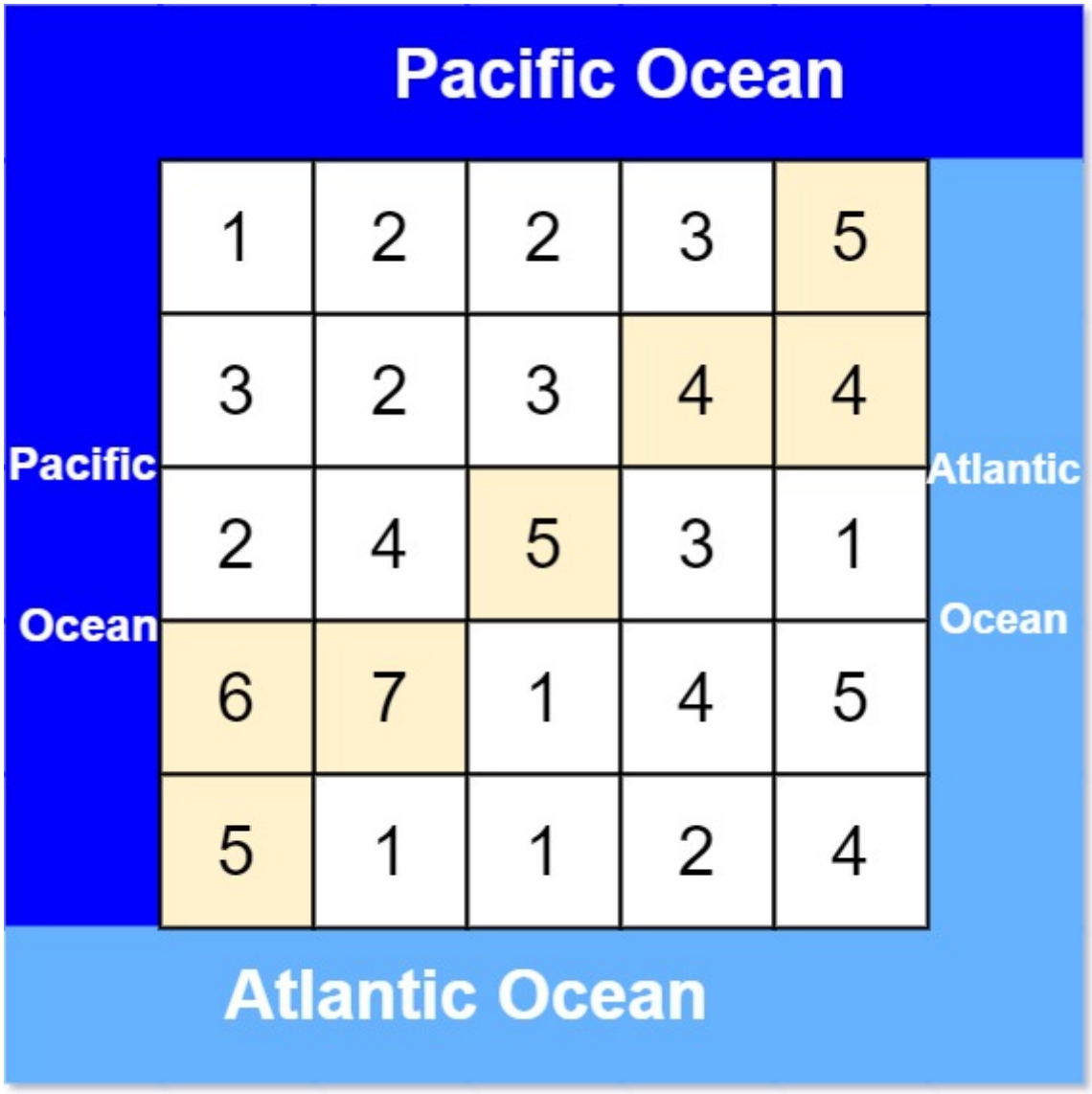
有一个 `m × n` 的矩形岛屿，与 **太平洋** 和 **大西洋** 相邻。 **“太平洋”** 处于大陆的左边界和上边界，而 **“大西洋”** 处于大陆的右边界和下边界。

这个岛被分割成一个由若干方形单元格组成的网格。给定一个 `m x n` 的整数矩阵 `heights` ， `heights[r][c]` 表示坐标 `(r, c)` 上单元格 **高于海平面的高度** 。

岛上雨水较多，如果相邻单元格的高度 **小于或等于** 当前单元格的高度，雨水可以直接向北、南、东、西流向相邻单元格。水可以从海洋附近的任何单元格流入海洋。

返回网格坐标 `result` 的 **2D 列表** ，其中 `result[i] = [ri, ci]` 表示雨水从单元格 `(ri, ci)` 流动 **既可流向太平洋也可流向大西洋** 。

示例 **1**：




```
输入: heights = [[1,2,2,3,5],[3,2,3,4,4],[2,4,5,3,1],[6,7,1,4,5],[5,1,1,2,4]]
输出: [[0,4],[1,3],[1,4],[2,2],[3,0],[3,1],[4,0]]
```

示例 **2**：

```
输入: heights = [[2,1],[1,2]]
输出: [[0,0],[0,1],[1,0],[1,1]]
```

提示：

- `m == heights.length`
- `n == heights[r].length`
- `1 <= m, n <= 200`
- `0 <= heights[r][c] <= 105`