0095. 不同的二叉搜索树 II

ITCharge 大约 2 分钟

- 标签：树、二叉搜索树、动态规划、回溯、二叉树
- 难度：中等

题目链接

- 0095. 不同的二叉搜索树 II - 力扣

题目大意

描述：给定一个整数 n。

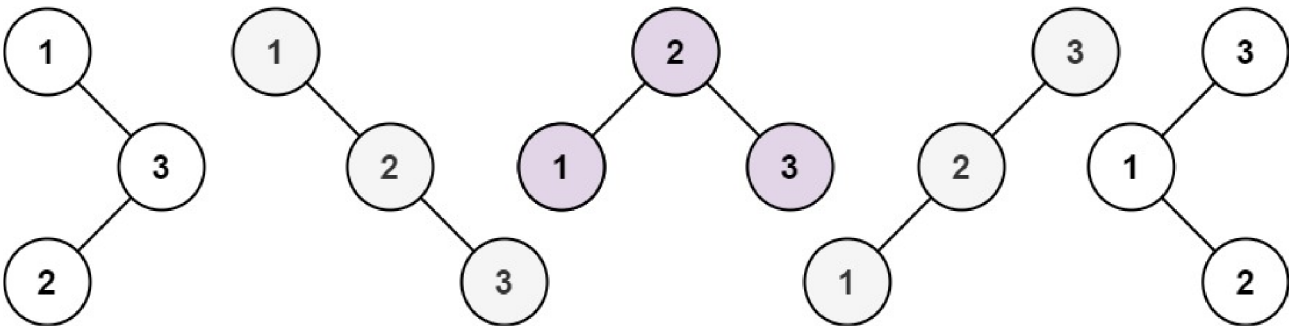
要求：请生成返回以 1 到 n 为节点构成的「二叉搜索树」，可以按任意顺序返回答案。

说明：

- $1 \leq n \leq 8$。

示例：

- 示例 1：




```py
输入：n = 3
输出：[[1,null,2,null,3],[1,null,3,2],[2,1,3],[3,1,null,null,2],[3,2,null,1]]
```

- 示例 2：

```py
输入：n = 1
输出：[[1]]
```

解题思路

思路 1：递归遍历

如果根节点为 i，则左子树的节点为 $(1, 2, ..., i - 1)$，右子树的节点为 $(i + 1, i + 2, ..., n)$。可以递归的构建二叉树。

定义递归函数 `generateTrees(start, end)` ，表示生成 $[left, ..., right]$ 构成的所有可能的二叉搜索树。

- 如果 $start > end$，返回 `[None]` 。
- 初始化存放所有可能二叉搜索树的数组。
- 遍历 $[left, ..., right]$ 的每一个节点 i，将其作为根节点。
  - 递归构建左右子树。
  - 将所有符合要求的左右子树组合起来，将其加入到存放二叉搜索树的数组中。
- 返回存放二叉搜索树的数组。

思路 1：代码

```py
class Solution:
    def generateTrees(self, n: int) -> List[TreeNode]:
        if n == 0:
            return []

        def generateTrees(start, end):
            if start > end:
                return [None]
            trees = []
            for i in range(start, end+1):
                left_trees = generateTrees(start, i - 1)
                right_trees = generateTrees(i + 1, end)
                for left_tree in left_trees:
                    for right_tree in right_trees:
                        curr_tree = TreeNode(i)
```

```
                    curr_tree.left = left_tree
                    curr_tree.right = right_tree
                    trees.append(curr_tree)
        return trees
    return generateTrees(1, n)
```

思路 1：复杂度分析

- **时间复杂度**：$O(C_n)$，其中 C_n 是第 n 个卡特兰数。
- **空间复杂度**：$O(C_n)$，其中 C_n 是第 n 个卡特兰数。