二叉树上的遍历法

遍历（Traversal），顾名思义，就是通过某种顺序，一个一个访问一个数据结构中的元素。比如我们如果需要遍历一个数组，无非就是要么从前往后，要么从后往前遍历。但是对于一棵二叉树来说，他就有很多种方式进行遍历：

1. 层序遍历（Level order）
2. 先序遍历（Pre order）
3. 中序遍历（In order）
4. 后序遍历（Post order）

- 分治法和遍历法

分治法（Divide & Conquer）与遍历法（Traverse）是两种常见的递归（Recursion）方法。

分治法解决问题的思路

先让左右子树去解决同样的问题，然后得到结果之后，再整合为整棵树的结果。

遍历法解决问题的思路

通过前序/中序/后序的某种遍历，游走整棵树，通过一个全局变量或者传递的参数来记录这个过程中所遇到的点和需要计算的结果。

两种方法的区别

从程序实现角度分治法的递归函数，通常有一个返回值，遍历法通常没有。

- **什么是回溯(Backtracking)?**

  有的时候，深度优先搜索算法（DFS），又被称之为回溯法，所以你可以完全认为回溯法，就是深度优先搜索算法。在我的理解中，回溯实际上是深度优先搜索过程中的一个步骤。比如我们在进行全子集问题的搜索时，假如当前的集合是 {1,2} 代表我正在寻找以 {1,2}开头的所有集合。那么他的下一步，会去寻找 {1,2,3}开头的所有集合，然后当我们找完所有以 {1,2,3} 开头的集合时，我们需要把 3 从集合中删掉，回到 {1,2}。然后再把 4 放进去，寻找以 {1,2,4} 开头的所有集合。这个把 3 删掉回到 {1,2} 的过程，就是回溯。

- 递归的三要素
  1.递归的定义
  2.递归的拆解（e.g 前序遍历中， 先记录根节点，再dfs左子节点， 再dfs右子节点）