Ww03:

递归的实现、特性和思维要点

首先还是要重复一遍：我们的程序到最后实际上执行的是有if..else.., for / while loop

递归：

本质是循环，通过函数体进行的循环；

向下进入到一层，向上回到原来一层；

每一层的环境和周围的人都是一份拷贝（局部的，不会互相影响），主角穿越到不同层次中（会发生变化->函数的参数）

递归最典型的例子：Fibonacci数列 和 n!

Python代码模板->记住：



3个思维要点：

不要人肉递归（最大误区 ）

找到最近最简方法，将其拆解成可重复解决 的问题（重复子问题，找最近重复性）

数学归纳法思维

实战：

爬楼梯问题：

根据给出条件反向推导出递归公式：只能走1步或2步

当n时，如果站在n-1上，走一步到n；如果站在n-2上，直接走2步到n；

* F(n) = f(n-1) + f(n-2)

考虑会不会有漏/重复

括号生成问题：

可以先把所有情况生成出来，然后输出之前筛选有效的括号

生成的时候就加判断条件：1)left未超标就可以放左括号 2)left >= right才可以放右括号

括号生成问题给后面很多问题提供了一个思路，就是给出n个格子，这个格子里面可以放左括号，也可以放右括号。。。

判断二叉搜索树问题：

递归（不能只递归比较左右结点，要递归左子树和右子树）

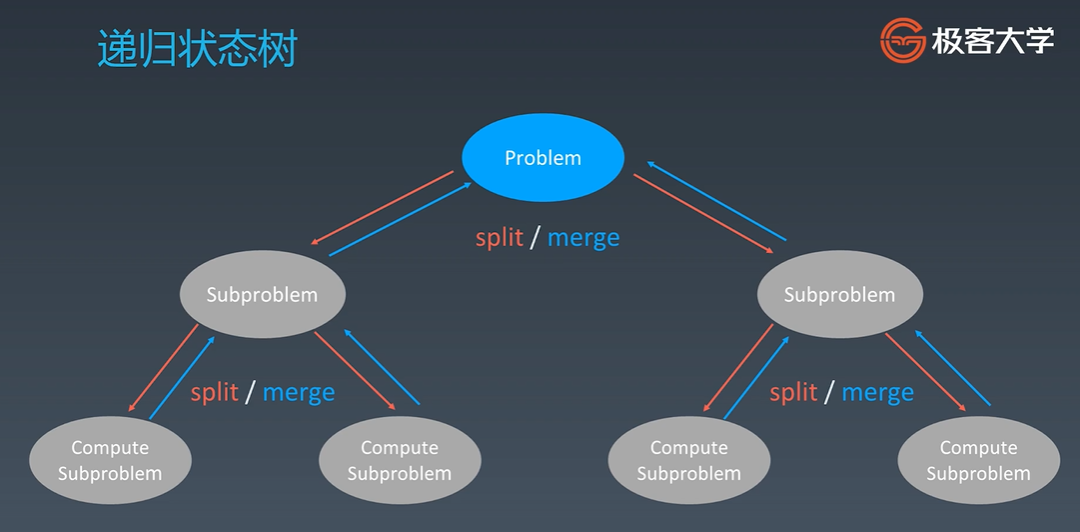
中序遍历是递增数列，遍历过程中判断当前元素是否大于等于前一个元素即可

分治(divide and conquer)和回溯(backtracking)

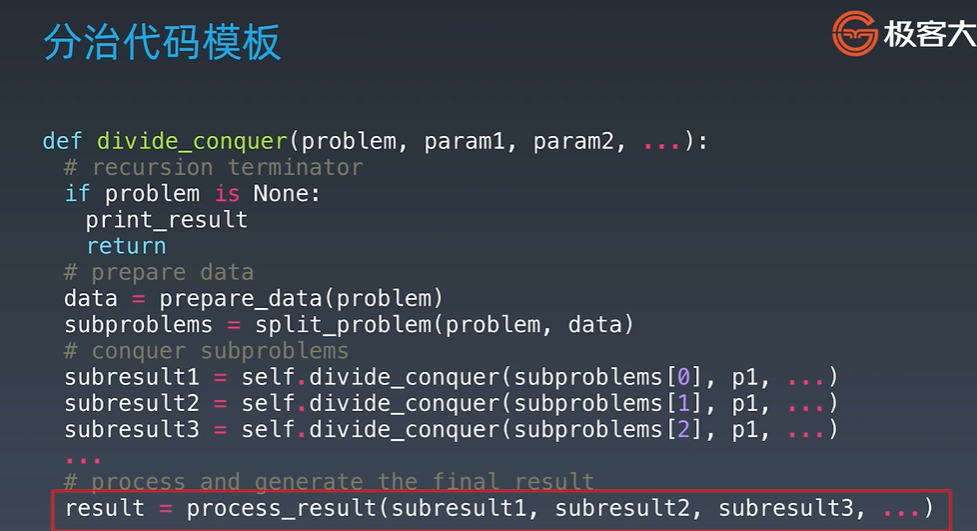
本质上是递归，特殊的递归

遇到一个问题找它的重复性，最优重复性->动态规划；最近重复性->一般的递归，分治，回溯

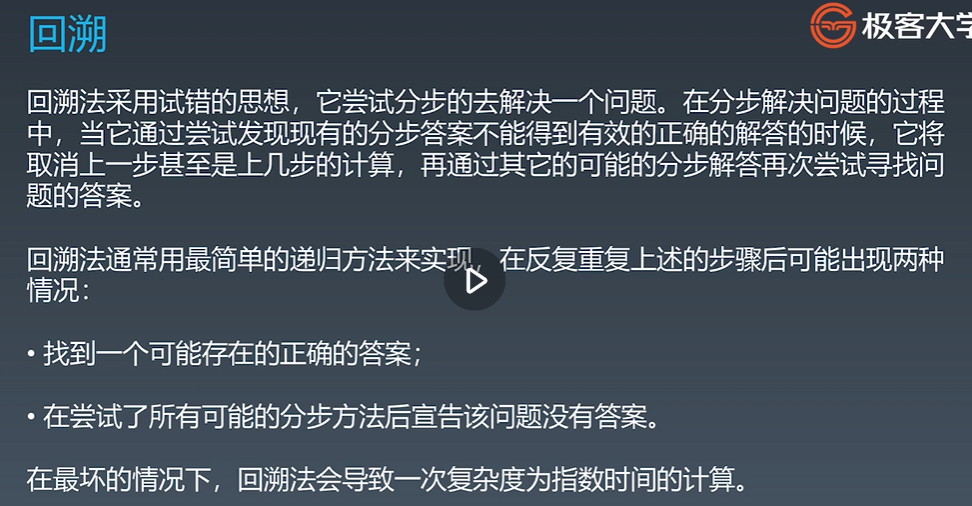
分治算法的思想：



分支代码模板：



回溯：不断的在每一层去试

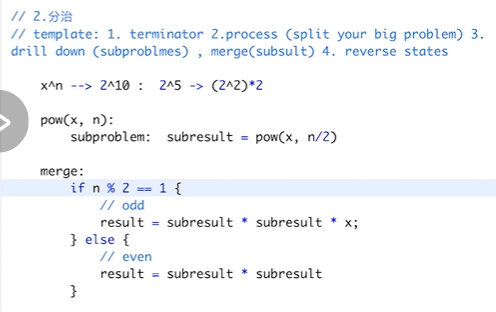


最典型的应用是处理八皇后问题和数组问题

实战：Pow(x, n)

暴力法：for循环 O(n)

分治法： O(logN)



**子集问题**：

分治法

迭代法\*

众数

电话号码的字母组合

分治

N皇后问题

分治