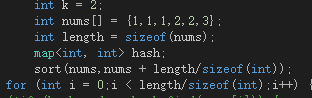
数组名字.sizeof();

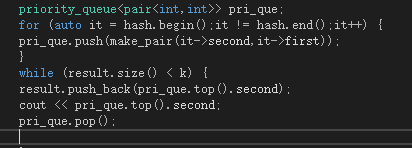


而vecotr的nums.size()直接就是向量长度

map<int, int> hash;

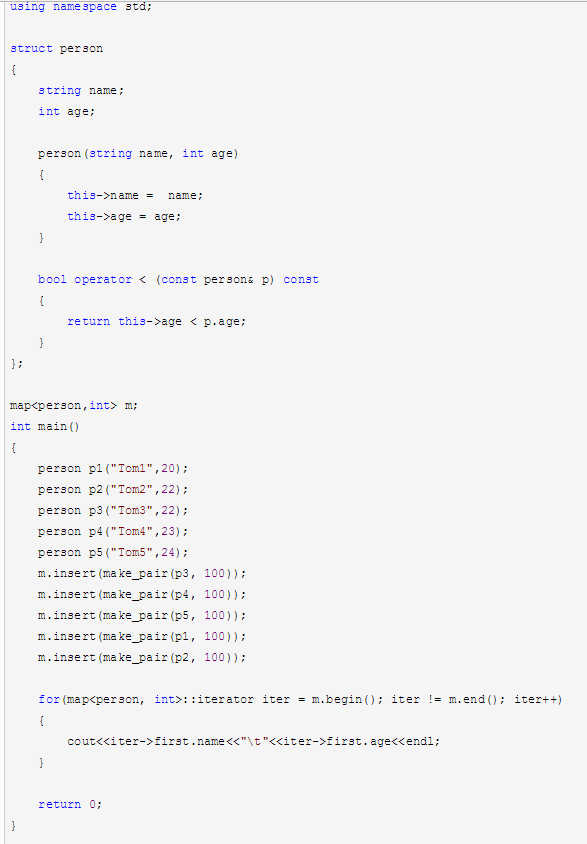
hash[nums[i]]++;

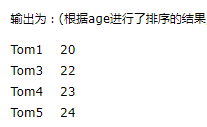
是给hash表键为nums[i]的内容加一



使用priority\_queue建大顶堆，这个pair<int,int>是存储first,second的。

当要插入键和值时，make\_pair( , )是要用的，map里面插入也可以用这个

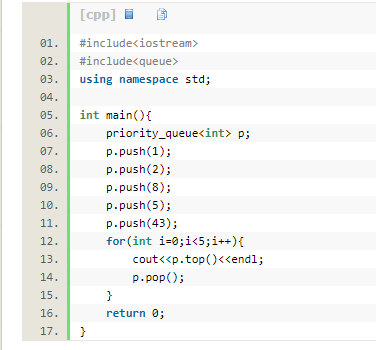


结果排序了

现在迭代器在C++11新标准里直接写成auto iter = m.begin()

map是根据内部元素排序不是根据键，而unordered\_map不排序

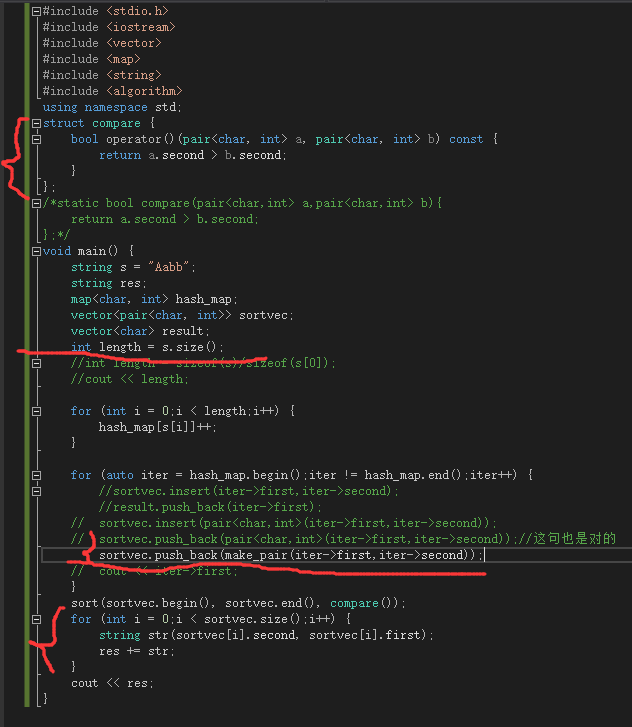
下图也是





用cout输出string字符串要在头文件里面加#include <string>

sort不能直接对map进行，要把map赋值给vector再sort,

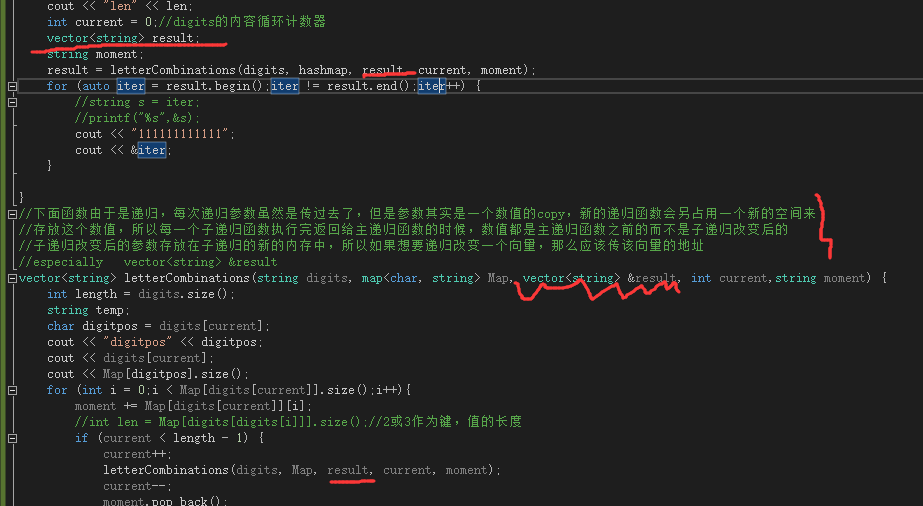


sort对vector是sort(nums.begin(),nums.end())

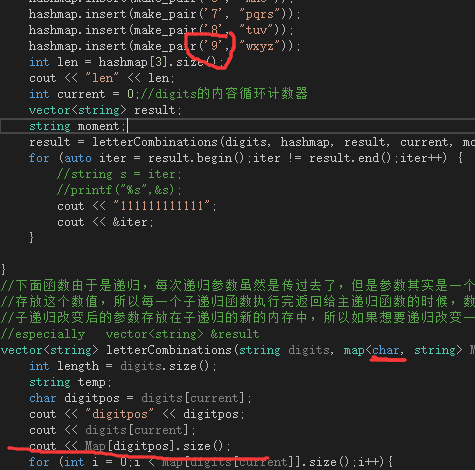
对数组是sort(nums,nums + length/sizeof(int));

一对插入用make\_pair

String s一串字符串，s[i]为char类型，不是string

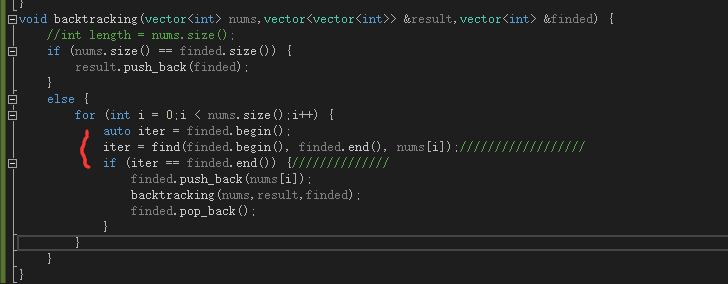


想要递归函数改变一个vector的值,递归函数传的是copy不是原变量，是变量的值的copy，赋给新的地址.

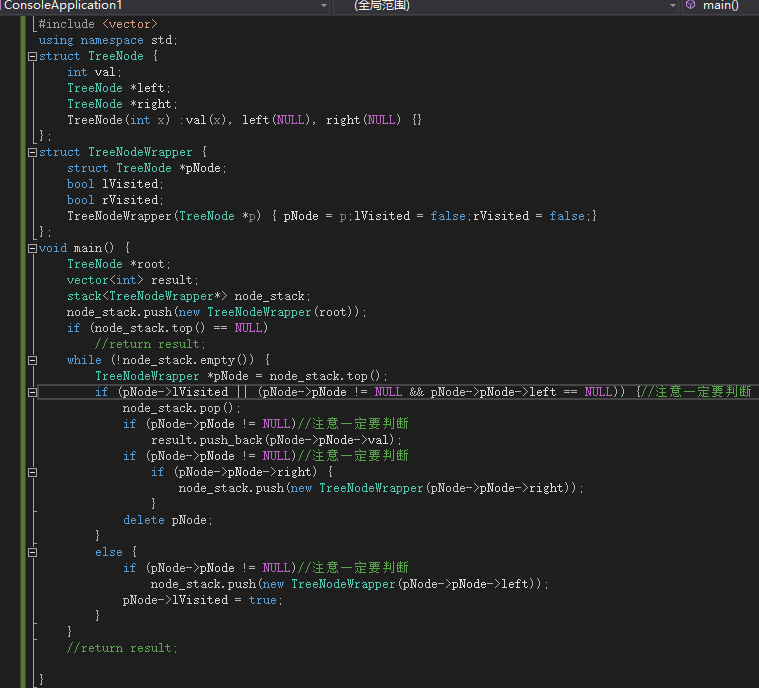


是char变量就要加两个单引号，不然map会以为他是int，虽然他定义为char,如果不加单引号那么读的时候会有问题

好好理解backtracking 17,这个递归函数



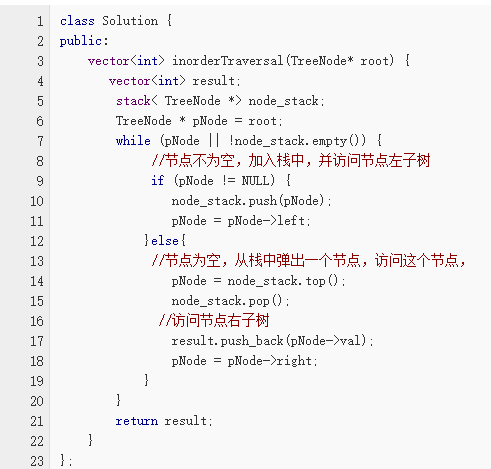
注意，这里vector的查找不能用finded.find(),这是map才可以用的，只能用头文件algorithm里面的find函数这样做，把查找位置返回给vector的迭代器。这个函数是一个递归的套路

这是二叉树的中序遍历，注意里面判断非空的条件!!!!!!!

<http://blog.csdn.net/sunao2002002/article/details/46315145>

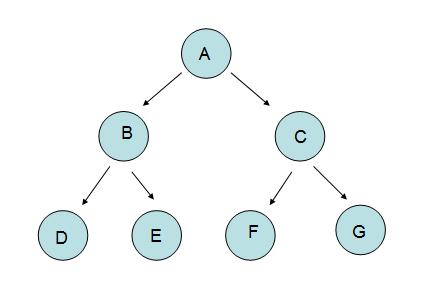
中序的两种方法非常有用

中序还有一种思想



就是先把根节点入栈，再把根节点左子树木入栈，再入栈左子树的左子树，知道没有左子树，就把栈顶弹出，即最左的叶子，每次弹出就会把top指向上一个父节点，再把右子树入栈，

把右子树的左子树入栈，就是说如果有左子树就入栈左，没有就弹出节点，入栈右子树，



中序是D,B,D,A,F,C,G,比如先入栈A,B,D，到D没有左子，就把D弹出入栈D右子，没有就把pNode指向现在的栈顶B，就弹出B，搜索B右子，把E入栈，E没有左子，弹出E，找到E右为空，于是弹出A，栈顶为A，E没有右子，pNode为A为栈顶，A右子C入栈，F入栈，指向F左孩子，为空，F弹出，访问F右孩子，没有，C弹出，G。

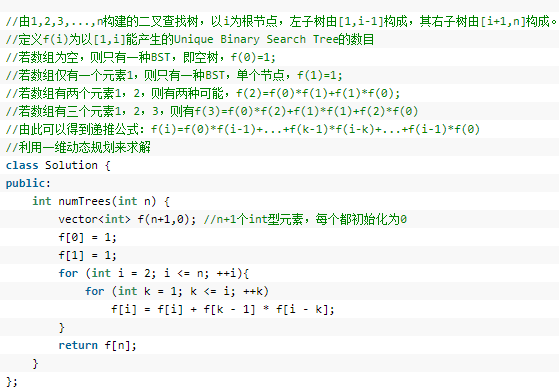
即一个结点a'a入栈之后把pNode指向它的左孩子，如果左孩子不为空继续向下，如果为空，马上弹出 a'a ，弹出aa就马上把pNode指向aa的右孩子，不为空就继续向下，为空就马上弹出aa的父亲bb。

即节点左孩子为空弹出节点马上入栈右节点反正仔细理解

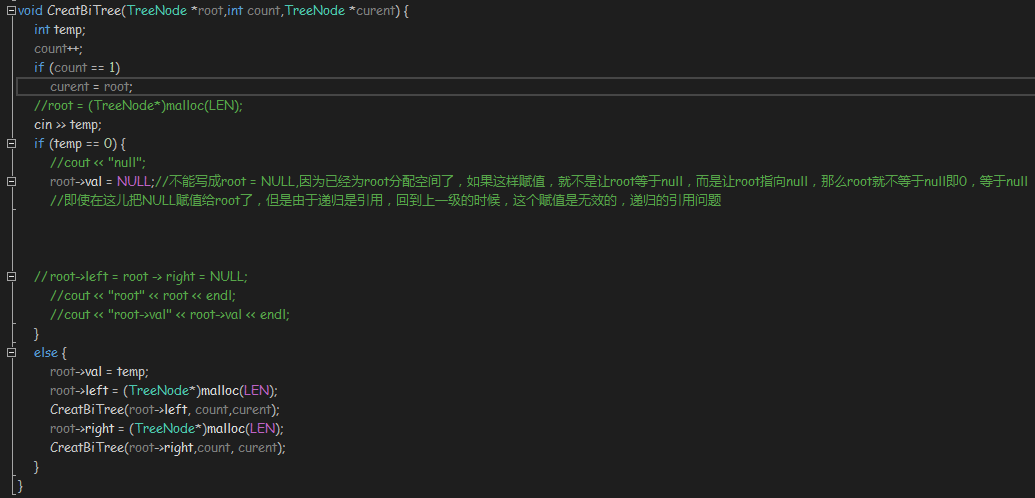


96题目





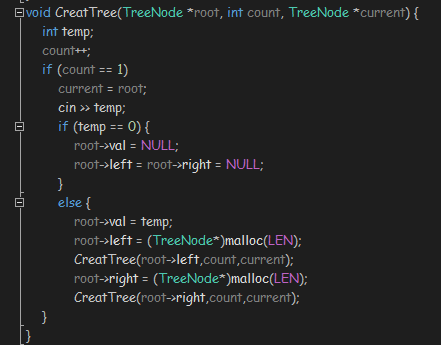
树的初始化

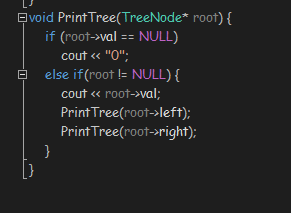


赋值进去的root要提前申请内存，

特别注意的是，因为这是提前给每个申请了内存，不管他是不是0，他的内存已经存在，如果把null赋值给他，是让他指向null这个空间，但是他是在内层递归里面进行的，所以一旦退出这个递归，那么他的值又复原了，递归只是改变了引用并不改变他本身，所以在这种情况下应该把这个节点的值赋为null。

如果真想把一个结点赋值为空，下面是另一种写法



这是打印