1. 代码题

class Solution {

public:

bool hasPathSum(TreeNode \*root, int sum) {

if (root == nullptr) {

return false;

}

if (root->left == nullptr && root->right == nullptr) {

return sum == root->val;

}

return hasPathSum(root->left, sum - root->val) ||

hasPathSum(root->right, sum - root->val);

}

};

回溯法打印二叉树的所有路径

class Solution {

public:

vector<string> res;

void backtrack(TreeNode \*root, string track)

{

If(root==NULL) return;

If(root->left==NULL&&root->right==NULL){

track+=to\_string(root->val);

res.push\_back(track);

return;

}

track+=to\_string(root->val);

track+=”->”;

backtrack(root->left,track);

backtrack(root->right,track);

}

Vector<string> binaryTreePaths(TreeNode \*root){

String track;

Cout<<track<<endl;

Backtrack(root,track);

Cout<<track<<endl;

Return res;

}

};

2. 知识点

什么是死锁？死锁产生的条件？

1). 死锁的概念

　　在两个或者多个并发进程中，如果每个进程持有某种资源而又等待其它进程释放它或它们现在保持着的资源，在未改变这种状态之前都不能向前推进，称这一组进程产生了死锁。通俗的讲，就是两个或多个进程无限期的阻塞、相互等待的一种状态。

2). 死锁产生的四个必要条件

互斥：至少有一个资源必须属于非共享模式，即一次只能被一个进程使用；若其他申请使用该资源，那么申请进程必须等到该资源被释放为止；

占有并等待：一个进程必须占有至少一个资源，并等待另一个资源，而该资源为其他进程所占有；

非抢占：进程不能被抢占，即资源只能被进程在完成任务后自愿释放

循环等待：若干进程之间形成一种头尾相接的环形等待资源关系

3). 死锁的处理基本策略和常用方法

　　解决死锁的基本方法主要有 预防死锁、避免死锁、检测死锁、解除死锁等。

　　(1). 死锁预防

　　死锁预防的基本思想是 只要确保死锁发生的四个必要条件中至少有一个不成立，就能预防死锁的发生，具体方法包括：

打破互斥条件：允许进程同时访问某些资源。但是，有些资源是不能被多个进程所共享的，这是由资源本身属性所决定的，因此，这种办法通常并无实用价值。

打破占有并等待条件：可以实行资源预先分配策略(进程在运行前一次性向系统申请它所需要的全部资源，若所需全部资源得不到满足，则不分配任何资源，此进程暂不运行；只有当系统能满足当前进程所需的全部资源时，才一次性将所申请资源全部分配给该线程)或者只允许进程在没有占用资源时才可以申请资源（一个进程可申请一些资源并使用它们，但是在当前进程申请更多资源之前，它必须全部释放当前所占有的资源）。但是这种策略也存在一些缺点：在很多情况下，无法预知一个进程执行前所需的全部资源，因为进程是动态执行的，不可预知的；同时，会降低资源利用率，导致降低了进程的并发性。

打破非抢占条件：允许进程强行从占有者那里夺取某些资源。也就是说，但一个进程占有了一部分资源，在其申请新的资源且得不到满足时，它必须释放所有占有的资源以便让其它线程使用。这种预防死锁的方式实现起来困难，会降低系统性能。

打破循环等待条件：实行资源有序分配策略。对所有资源排序编号，所有进程对资源的请求必须严格按资源序号递增的顺序提出，即只有占用了小号资源才能申请大号资源，这样就不回产生环路，预防死锁的发生。

　　(2). 死锁避免的基本思想

　　死锁避免的基本思想是动态地检测资源分配状态，以确保循环等待条件不成立，从而确保系统处于安全状态。所谓安全状态是指：如果系统能按某个顺序为每个进程分配资源（不超过其最大值），那么系统状态是安全的，换句话说就是，如果存在一个安全序列，那么系统处于安全状态。资源分配图算法和银行家算法是两种经典的死锁避免的算法，其可以确保系统始终处于安全状态。其中，资源分配图算法应用场景为每种资源类型只有一个实例(申请边，分配边，需求边，不形成环才允许分配)，而银行家算法应用于每种资源类型可以有多个实例的场景。

　　(3). 死锁解除

　　死锁解除的常用两种方法为进程终止和资源抢占。所谓进程终止是指简单地终止一个或多个进程以打破循环等待，包括两种方式：终止所有死锁进程和一次只终止一个进程直到取消死锁循环为止；所谓资源抢占是指从一个或多个死锁进程那里抢占一个或多个资源，此时必须考虑三个问题：

　　(I). 选择一个牺牲品

　　(II). 回滚：回滚到安全状态

　　(III). 饥饿（在代价因素中加上回滚次数，回滚的越多则越不可能继续被作为牺牲品，避免一个进程总是被回滚）

3. 开源项目

学习了github的搜索技巧

（1）按项目名称查找 in：name

（2）按项目描述查找 in：description

（3）按README描述查找 in：readme

（4）设置星标数查询范围 stars

（5）设置fork数查询范围 forks

（6）按项目语言筛选 language

（7）按项目作者查找 user

（8）按项目大小查找 size

（9）组合查找 in：name xxx in：description xxx in：readme xxx