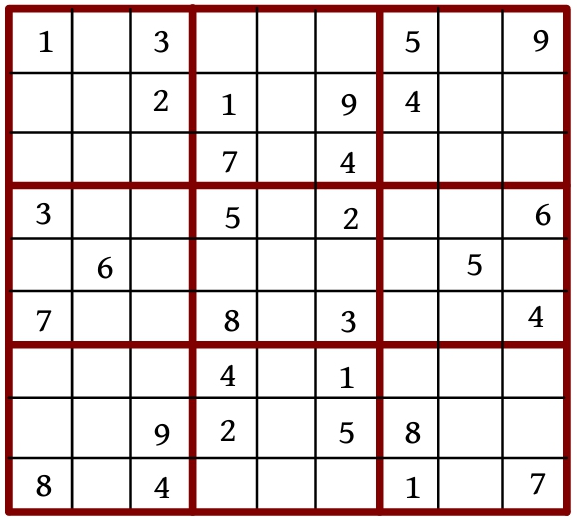
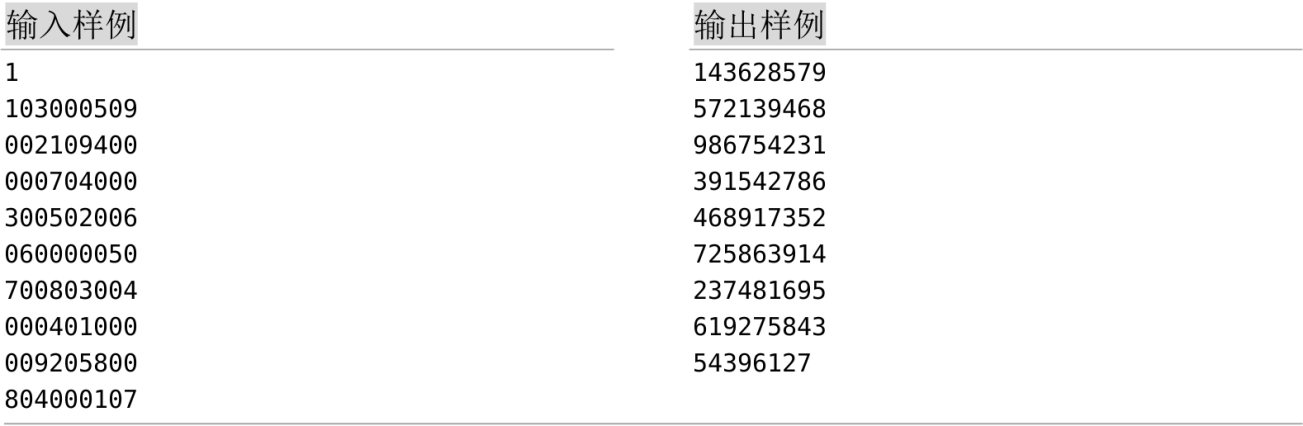
**POJ2676**

**题目描述（POJ2676）：**数独是一项非常简单的任务。如下图所示，一张9行9列的表被分成9个3×3的小方格。在一些单元格中写上十进制数字1～9，其他单元格为空。目标是用1～9的数字填充空单元格，每个单元格一个数字，这样在每行、每列和每个被标记为3×3的子正方形内，所有1～9的数字都会出现。编写一个程序来解决给定的数独任务。



**输入：**输入数据将从测试用例的数量开始。对于每个测试用例，后面都跟9行，对应表的行。在每一行上都给出9个十进制数字，对应这一行中的单元格。如果单元格为空，则用0表示。

**输出：**对于每个测试用例，程序都应该以与输入数据相同的格式打印解决方案。空单元格必须按照规则填充。如果解决方案不是唯一的，那么程序可以打印其中任何一个。



**题解：**本题为数独游戏，为典型的**九宫格问题，可以采用回溯法搜索**。把一个9行9列的网格再细分为9个3×3的子网格，要求在每行、每列、每个子网格内都只能使用一次1～9的一个数字，即在每行、每列、每个子网格内都不允许出现相同的数字。

0表示空白位置，其他均为已填入的数字。要求填完九宫格并输出（如果有多种结果，则只需输出其中一种）。如果给定的九宫格无法按要求填出来，则输出原来所输入的未填的九宫格。

用3个数组标记每行、每列、每个子网格已用的数字。

• row[i][x]：用于标记第i行中的数字x是否出现。

• col[j][y]：用于标记第j列中的数字y是否出现。

• grid[k][z]：标记第k个3×3子网格中的数字z是否出现。

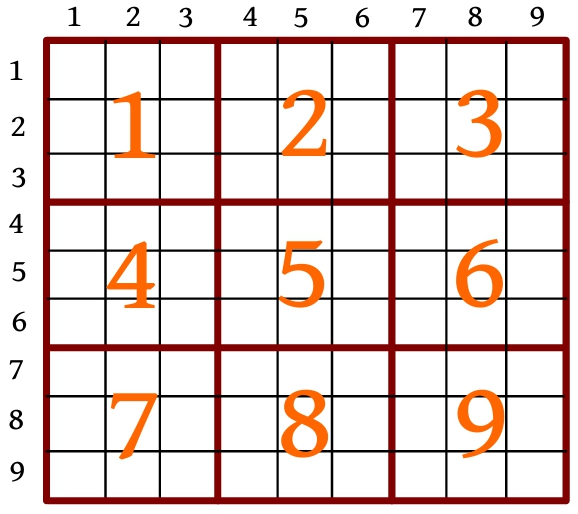
row和col的标记比较好处理，关键是找出grid子网格的序号与行i、列j的关系，即要知道第i行j列的数字属于哪个子网格。

把一个9行9列的网格再细分为9个3×3的子网格，在每个子网格内都不允许出现相同的数字，那么我们将9个子网格编号为1～9，在同一个子网格内不允许出现相同的数字。观察子网格的序号k与行i、列j的关系：

• 如果把第1～3行转换为0，第4～5行转换为1，第7～9行转换为2，则a=(i-1)/3；

• 如果把第1～3列转换为0，第4～5列转换为1，第7～9列转换为2，则b=(j-1)/3。

行i、列j对应的子网格编号k=3×a+b+1=3×((i-1)/3)+(j-1)/3+1，如下图所示。



**1. 算法设计**

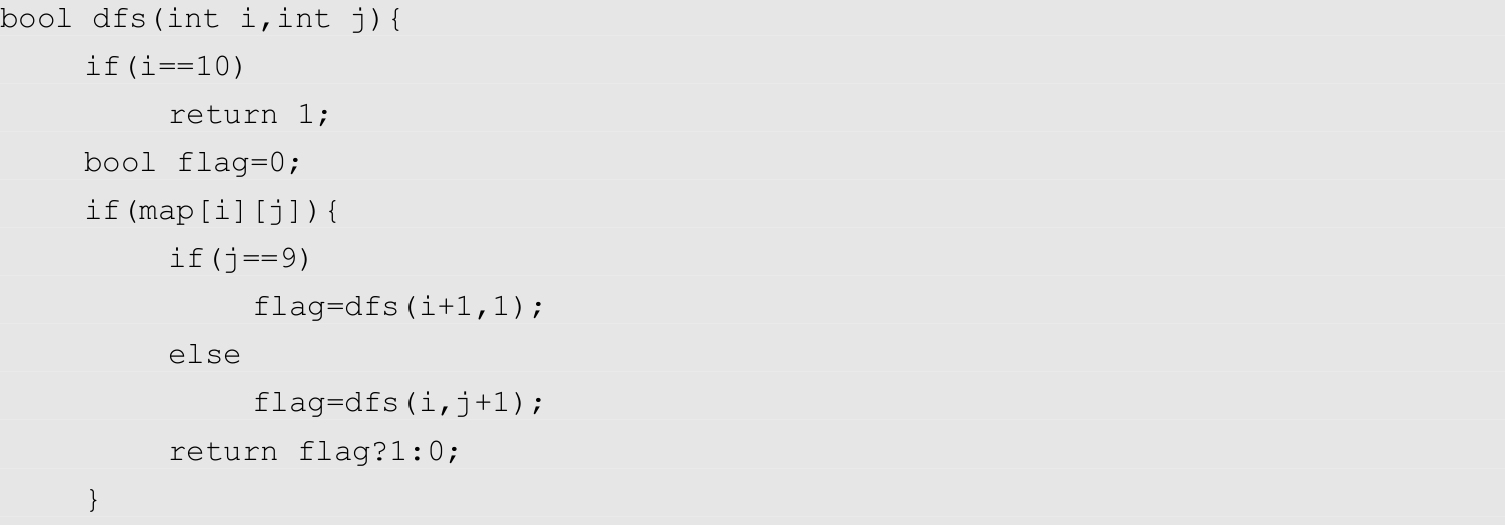
（1）预处理输入数据。

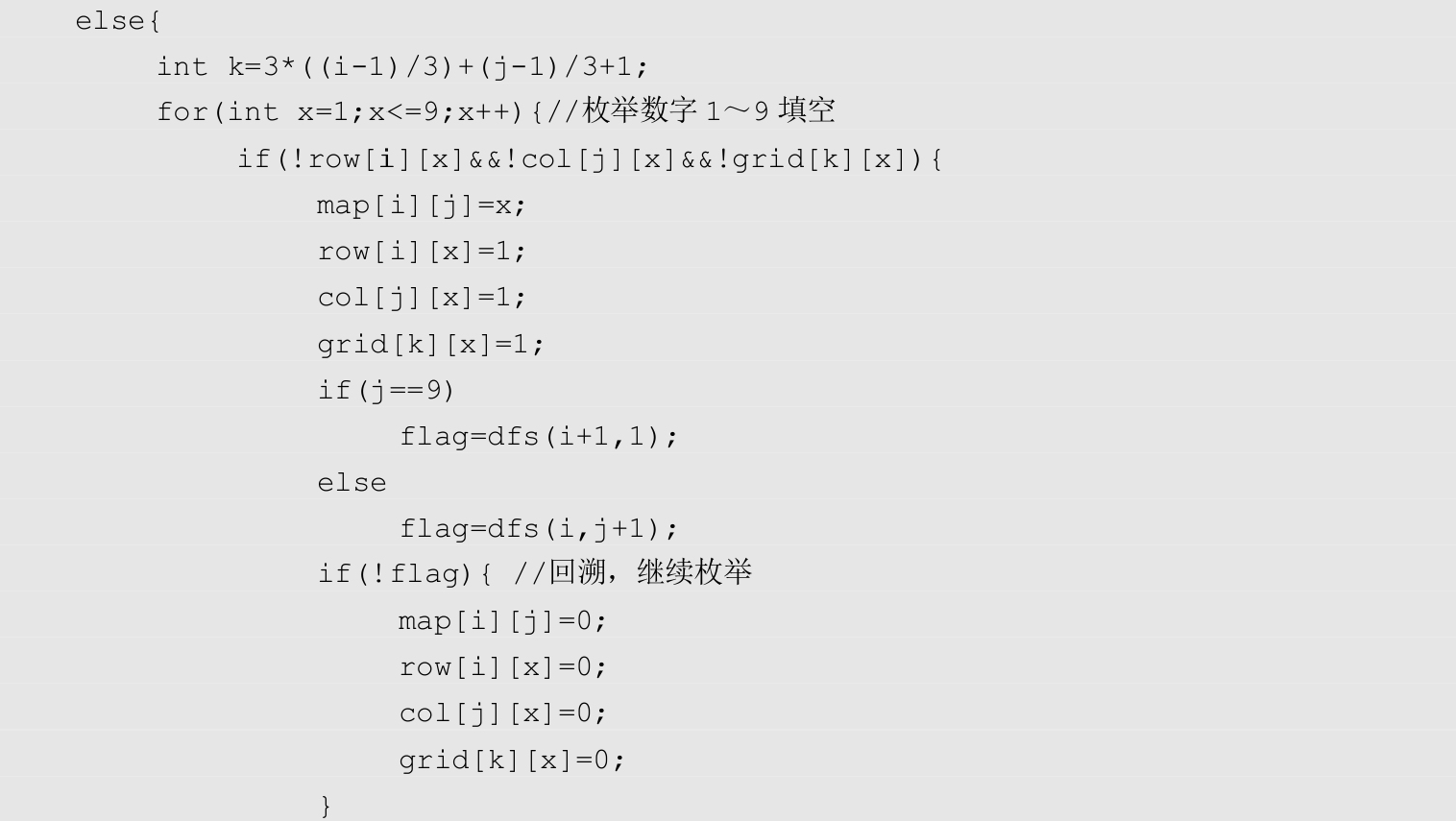
（2）从左上角(1,1)开始按行搜索，如果行i=10，则说明找到答案，返回1。

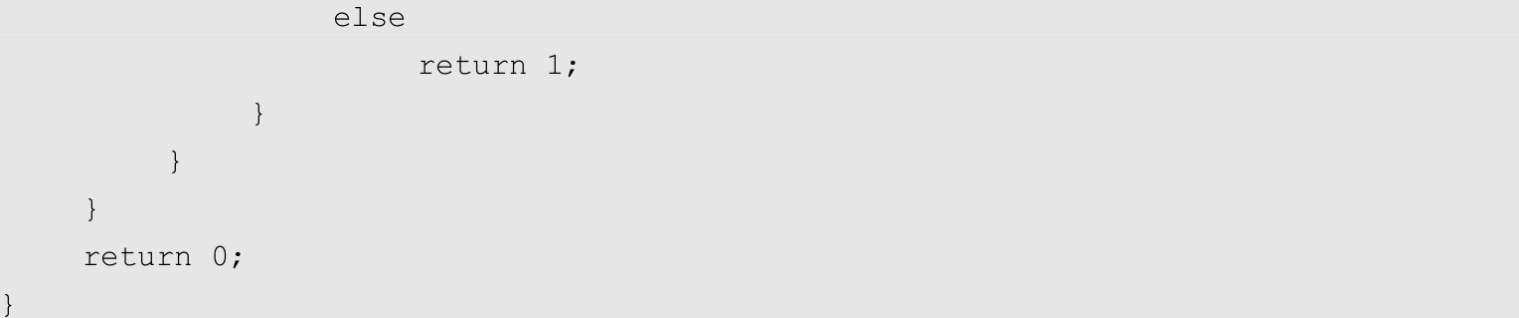
（3）如果map[i][j]已填数字，则判断如果列j=9，则说明处理到当前行的最后一列，继续下一行第1列的搜索，即dfs(i+1,1)，否则在当前行的下一列搜索，即dfs(i, j+1)。如果搜索成功，则返回1，否则返回0。

（4）如果map[i][j]未填数字，则计算当前位置(i,j)所属子网格k=3×((i-1)/3)+(j-1)/3+1。枚举数字1～9填空，如果当前行、当前列、当前子网格均未填该数字，则填写该数字并标记该数字已出现。如果判断列j=9，则说明处理到当前行的最后一列，继续下一行第1列的搜索，即dfs(i+1,1)，否则在当前行的下一列搜索，即dfs(i, j+1)。如果搜索失败，则回溯归位，继续搜索，否则返回1。

**2. 算法实现**

****

****

****

**POJ1190**

**题目描述（POJ1190）：**制作一个体积为Nπ的M层生日蛋糕，每层都是一个圆柱体。设从下往上数第i（1≤i≤M）层蛋糕是半径为Ri、高度为Hi的圆柱。当i<M时，要求Ri>Ri+1且Hi>Hi+1。由于要在蛋糕上抹奶油，所以为了尽可能节约经费，希望蛋糕外表面（底层的下底面除外）的面积Q最小。令Q=Sπ，对给出的N和M，找出蛋糕的制作方案（适当的Ri和Hi的值），使S最小。除Q外，以上所有数据皆为正整数。

**输入：**输入包含两行，第1行为N（N≤10 000），表示制作的蛋糕的体积为Nπ；第2行为M（M≤20），表示蛋糕的层数。

**输出：**单行输出一个正整数S（若无解，则S=0）。

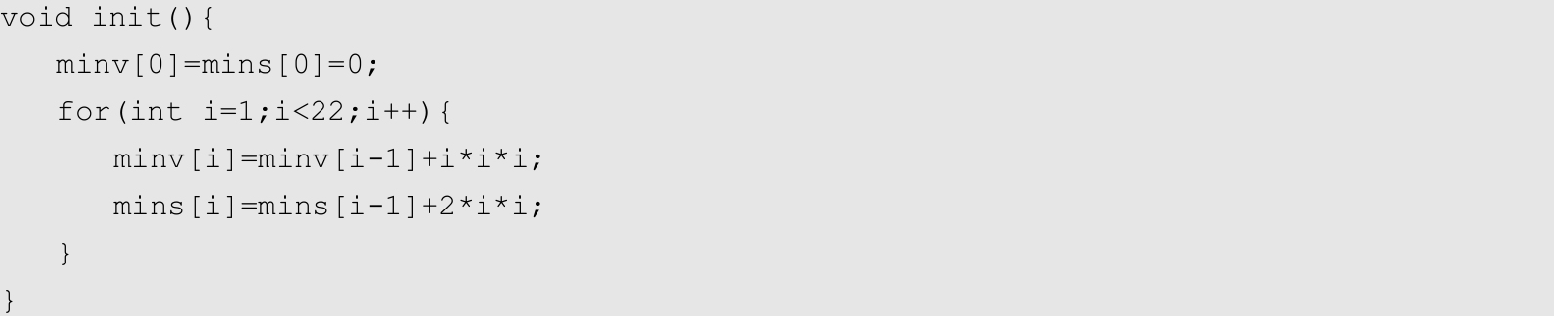


**提示：**圆柱体积V=πR2H，侧面积A'=2πRH，底面积A=πR2。

**题解：**本题为在体积和层数一定的情况下，找到合适的半径和高度，使蛋糕表面积最小。可以采用回溯法搜索求解。

**1. 预处理**

从顶层向下计算出最小体积和面积的最小可能值。在从顶层（即第1层）到第i层的最小体积minv[i]成立时，第i层的半径和高度都为i。此时只计算侧面积，对上表面积只在底层计算一次，底层的底面积即总的上表面积。



**2. 算法设计**

dep指当前深度；sumv、sums分别指当前体积和、面积和；r、h分别指当前层半径、高度。

**（1）从底层m层向上搜，当dep=0时，搜索完成，更新最小面积。**

**（2）剪枝技巧：**

• 如果当前体积加上剩余上面几层的最小体积大于总体积n，则退出；

• 如果当前面积加上剩余上面几层的最小面积大于最小面积，则退出；

• 如果当前面积加上剩余面积（剩余体积折算）大于最小面积，则退出。

**（3）枚举半径i，按递减顺序枚举dep层蛋糕半径的每一个可能值i，第dep层的半径最小值为dep。**

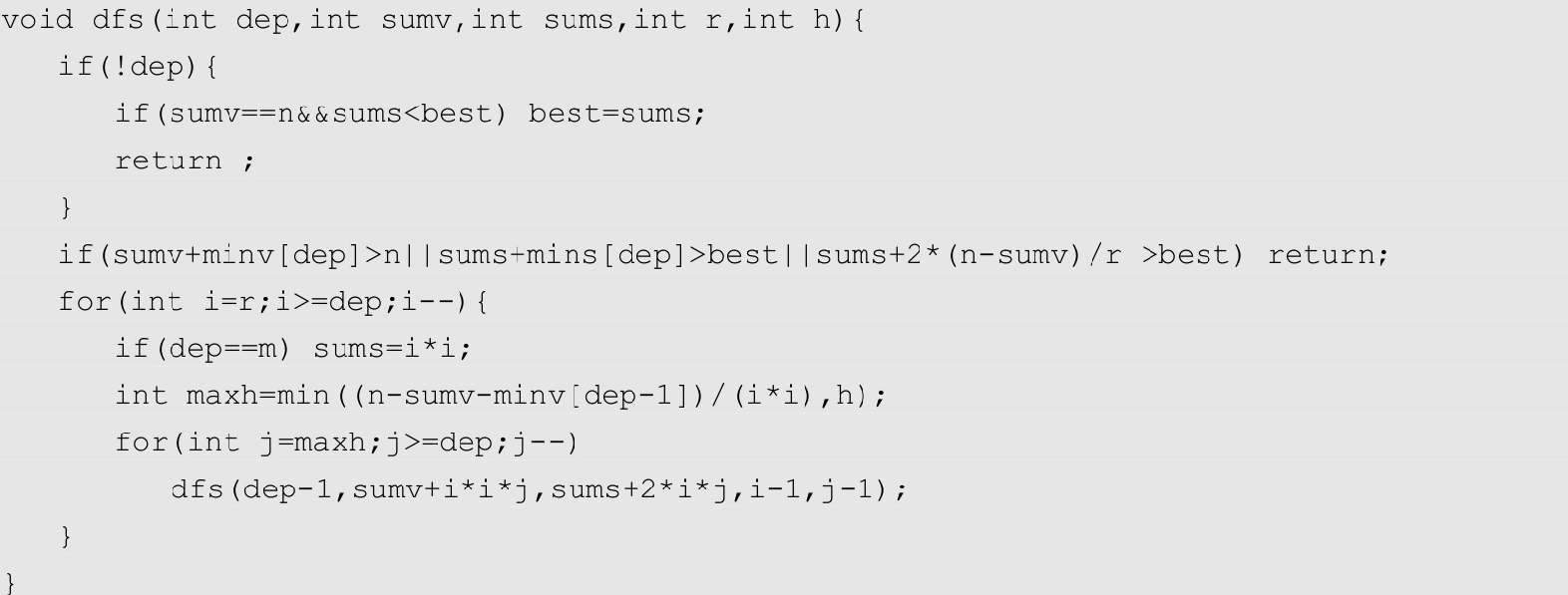
• 如果dep=m，sums=i×i，底面积作为外表面积的初始值（总的上表面积，以后只需计算侧面积）。

• 计算最大高度maxh，即dep层蛋糕高度的上限，(n-sumv-minv[dep-1])表示第dep层的最大体积。

• 枚举高度j，按递减顺序枚举dep层蛋糕高度的每一个可能值j，第dep层的最小高度值为dep。

• 递归搜索子状态，层次为dep-1，体积和为sumv+i×i×j，面积和为sums+2×i×j，半径为i-1，高度为j-1，即dfs(dep-1,sumv+i×i×j,sums+2×i×j,i-1,j-1)。

**3. 算法实现**

****

**说明：**

（1）**初始参数r和h均为n**。因为体积V=πR2H，因此体积为nπ时，n=R2H，半径和高度均不会超过n，**半径和高度均大于或等于当前层**。

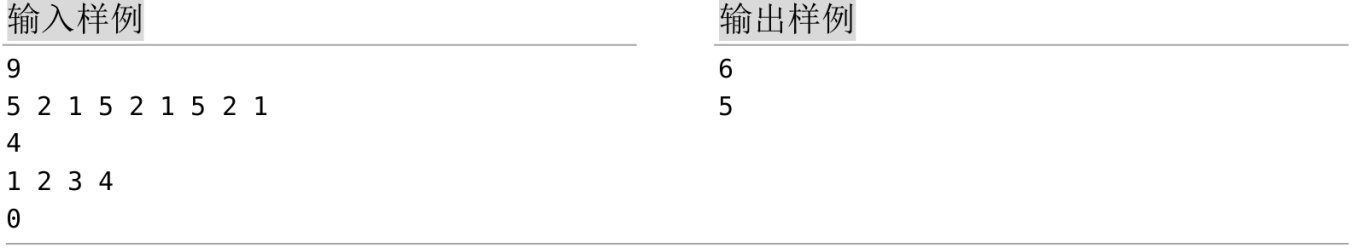
**（2）剩余面积折算。**体积V=πR2H，侧面积A'=2πRH，2V/R=A'，因此将剩余体积折算成剩余侧面积为2×(n-sumv)/r。

**POJ1011**

**题目描述（POJ1011）**：乔治拿来一组等长的木棒，将它们随机砍断，使得每一节木棒的长度都不超过50个长度单位。然后他又想把这些木棍恢复到原来的状态，但忘记了初始时有多少木棒及木棒的初始长度。请计算初始时原木棒的最小可能长度。每一节木棒的长度均为大于零的整数。

**输入：**输入包含多组数据，每组数据都包括两行。第1行是一个不超过64的整数，表示砍断之后共有多少节木棒。第2行是截断以后所得到的各节木棒的长度。在最后一组数据之后是一个0。

**输出：**对每组数据，都单行输出原木棒的最小长度。



**题解：**

**1. 算法设计**

本题由切割后的木棒长度推测原木棒的最小长度，可以枚举原木棒的最小长度，使用回溯法搜索及剪枝优化即可解决。可以用拼接的方法反向推测，根据现有木棒拼接成多个等长的原木棒。例如，1 2 3 4，最多可以拼接成两根等长木棒4+1、3+2，原木棒的最小长度为5。例如，5 2 1 5 2 1 5 2 1，最多可以拼接成4根等长木棒5+1、5+1、5+1、2+2+2，原木棒的最小长度为6。

**（1）枚举长度。**木棒的总长度为sumlen，最长木棒的长度为maxlen。因为切割后最长为maxlen，那么原木棒的长度必然大于或等于maxlen。如果原木棒只有一根，那么原木棒的长度就是sumlen。如果原木棒多于一根，那么原木棒的长度一定小于或等于sumlen/2。从maxlen到sumlen/2，从小到大枚举所有可能的原木棒长度，通过深度优先搜索尝试能否组合成原木棒，如果尝试成功，则当前木棒的长度为原木棒的最小可能长度。

**（2）组合顺序。**对**木棒长度从大到小排序**，如果**从小到大排序则会超时**。因为小木棒比大木棒灵活性更好，所以先考虑较长的木棒，然后用较短的木棒组合成原棒，更容易成功。好比往箱子装东西，尽量先装大的，然后用小的填补空隙，如果先把小的装进去，大的就可能放不下，或者装不满。用一维数组used[]标记当前状态下木棒是否已使用组合原棒。

**（3）剪枝技巧。**

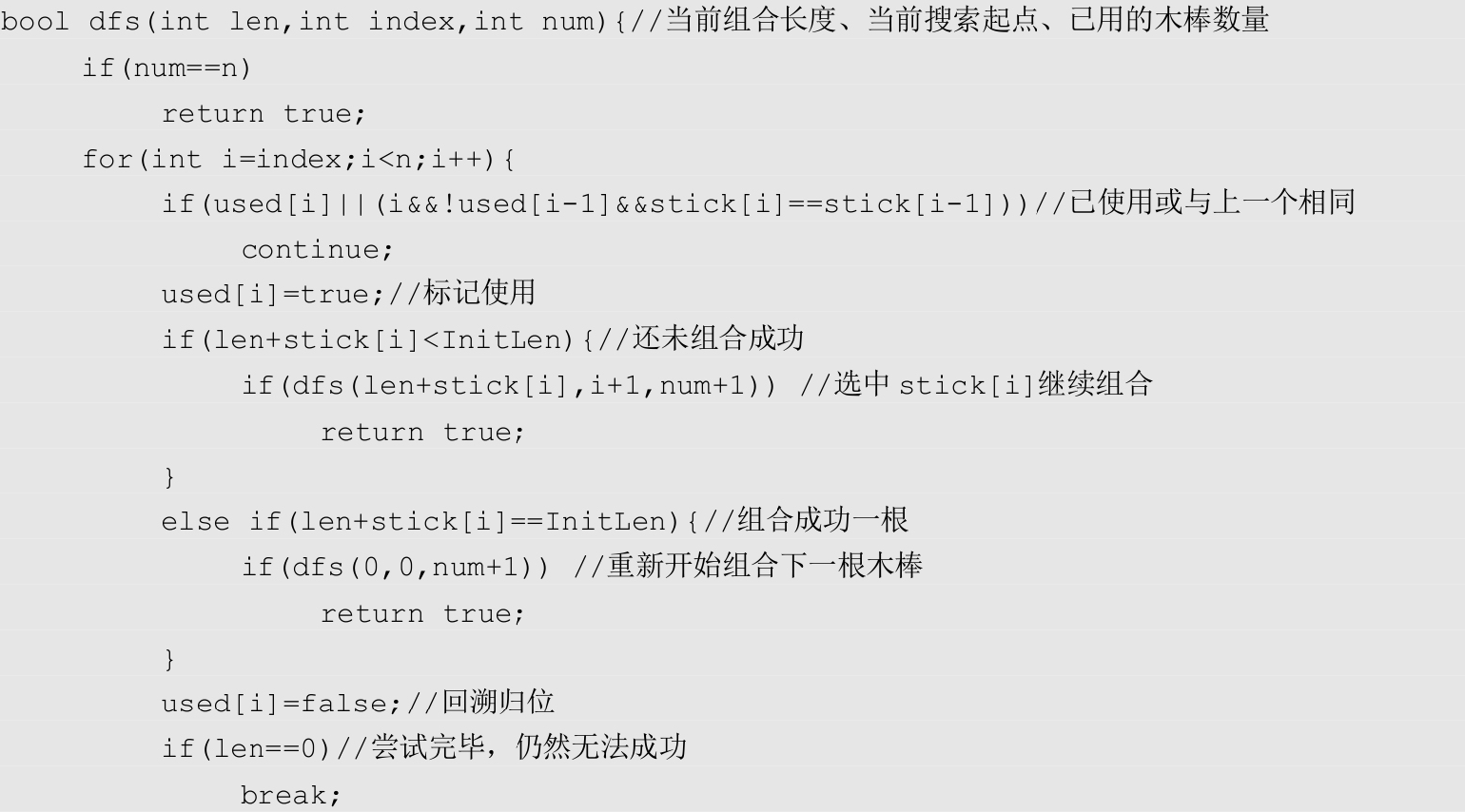
• 剪枝技巧1：从小到大枚举，第1个满足条件的原木棒长度InitLen必然是最短的。

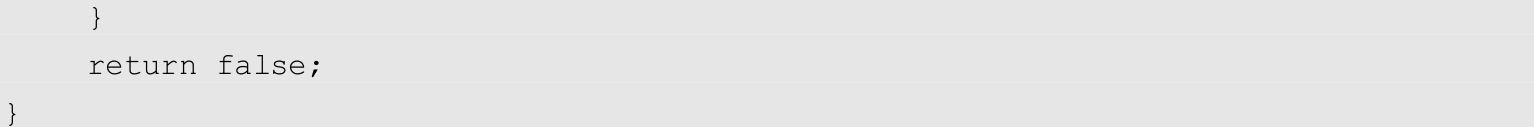
• 剪枝技巧2：原木棒是等长的，因此sumlen%InitLen=0。

• 剪枝技巧3：如果当前木棒已使用或者与前一个未使用的木棒长度相等，则无须再搜索。

• 剪枝技巧4：组合新木棒时，若搜索完所有木棒后都无法组合，则说明该木棒无法在当前组合方式下组合，不用往下搜索，直接返回上一层。

**2. 算法实现**

****

****