***4皇后问题就是求解如何在4×4的棋盘上无冲突的摆放4个皇后棋子。在任意一个皇后所在位置的水平、竖直、以及45度斜线上都不能出现皇后的棋子。***

***要求编程求出符合要求的情况的个数。***

***回溯法的基本思想是：可以构建出一棵解空间树，通过探索这棵解空间树，可以得到四皇后问题的一种或几种解。***

***在构建解空间树的途中进行深度优先探索，当探索到某一种棋盘局面一定不是四皇后问题的解的时候（比如出现任意两个或两个以上的棋子在同一行/同一列/45度斜线上），就可以判断这个节点向下派生出的解空间树的节点也一定不是四皇后问题的解，这样就可以避免大量的无用功。***

***这样，如果能够成功遍历到叶子节点，并且判断该叶子节点的局面就是符合4皇后问题的，那么这个节点局面就代表一个合法的四皇后问题的解。***

***#include <stdio.h>***

***int count = 0;***

***int isCorrect(int i, int j, int (\*Q)[4])***

***{***

***int s, t;***

***for(s=i,t=0; t<4; t++)***

***if(Q[s][t]==1 && t!=j)***

***return 0;//判断行***

***for(t=j,s=0; s<4; s++)***

***if(Q[s][t]==1 && s!=i)***

***return 0;//判断列***

***for(s=i-1,t=j-1; s>=0&&t>=0; s--,t--)***

***if(Q[s][t]==1)***

***return 0;//判断左上方***

***for(s=i+1,t=j+1; s<4&&t<4;s++,t++)***

***if(Q[s][t]==1)***

***return 0;//判断右下方***

***for(s=i-1,t=j+1; s>=0&&t<4; s--,t++)***

***if(Q[s][t]==1)***

***return 0;//判断右上方***

***for(s=i+1,t=j-1; s<4&&t>=0; s++,t--)***

***if(Q[s][t]==1)***

***return 0;//判断左下方***

***return 1;//否则返回***

***}***

***void Queue(int j, int (\*Q)[4])***

***{***

***int i,k;***

***if(j==4){//递归结束条件***

***for(i=0; i<4; i++){***

***//得到一个解，在屏幕上显示***

***for(k=0; k<4; k++)***

***printf("%d ", Q[i][k]);***

***printf("\n");***

***}***

***printf("\n");***

***count++;***

***return ;***

***}***

***for(i=0; i<4; i++){***

***if(isCorrect(i, j, Q)){//如果Q[i][j]可以放置皇后***

***Q[i][j]=1;//放置皇后***

***Queue(j+1, Q);//递归深度优先搜索解空间树***

***Q[i][j]=0;//这句代码就是实现回溯到上一层***

***}***

***}***

***}***

***int main()***

***{***

***int Q[4][4];***

***int i, j;***

***for(i=0; i<4; i++)***

***for(j=0; j<4; j++)***

***Q[i][j] = 0;***

***Queue(0, Q);***

***printf("The number of the answers are %d\n", count);***

***return 0;***

***}***