**四皇后问题的解决方法和代码**

**四皇后问题描述：**在 4\*4 的棋盘上无冲突的摆放 4 个皇后，无冲突是指一个皇后所在位置的水平、竖直以及斜线上不能出现其他的皇后，其他的 n 皇后问题以此类推

**四皇后问题的解决方法：**按行来摆放棋子，下一行的摆放满足于与上一行的棋子没有冲突，否则就返回上一步走其他的路线，这就是所谓的回溯法

**程序源代码：**

**#include<stdio.h>**

**#define N 4**

**char board[N][N];**

**int t;**

**int col[N];**

**int safetyPlace(int x,int y)**

**{**

**int i,j;**

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

**{**

**j = col[i];**

**if(x==i||y==j)**

**return 0;**

**if(x-y==i-j||x+y==i+j)**

**return 0;**

**}**

**return 1;**

**}**

**void get\_position(int i)**

**{**

**int w,j;**

**char a[1]={0};**

**if(i==N)**

**{**

**for(w=0;w<N;w++)**

**{**

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

**{**

**if(board[w][j]==001)**

**printf("%d ",board[w][j]);**

**else**

**{**

**printf("%d ",a[0]);**

**}**

**}**

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

**}**

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

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

**t++;**

**}**

**else**

**{ int u;**

**for (u=0;u<N;u++)**

**{ if(safetyPlace(i,u)==1)**

**{**

**col[i]=u;**

**board[i][u]=001;**

**get\_position(i+1);**

**col[i]=0;**

**board[i][u]=0;**

**}**

**}**

**}**

**}**

**main()**

**{**

**printf("%d是皇后!\n\n",001);**

**get\_position(0);**

**printf("一共有%d种方法!\n",t);**

**}**