**深度优先搜索算法的实现（java实现）**

**171491125 吴俊**

**一．问题：**

找出四皇后问题的全部解。

**二．实现思路：**

就是将所有的结果变成一棵树，从树的结点开始访问，采用深度优先策略，从树的根结点开始访问，如果满足条件，继续访问下一层，如果不满足条件，返回上一个结点，继续访问其它结点。重复操作。

**三．实验代码：**

**public** **class** huanghou {

**public** **static** **int** *Empress*=4; //代表四皇后

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

**int**[][] chessboard=**new** **int**[*Empress*][*Empress*];

**int** i,j;

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

**for**(j=0;j<*Empress*;j++) {

chessboard[i][j]=0;

}

}

*Empress\_solution*(0,chessboard);

}

**public** **static** **void** Empress\_solution(**int** a,**int**[][] chessboard) {

**if**(a==*Empress*){//递归结束条件(四层棋盘已经遍历完）

**for**(**int** i=0;i<*Empress*;i++)

{

**for**(**int** j=0;j<*Empress*;j++)

{

System.***out***.print(chessboard[i][j]+" ");

}

System.***out***.println("\n");

}

System.***out***.println("\*\*\*\*\*\*\*\*\*\*\*\*\*");

}

//递归计算,i代表行

**for**(**int** i=0;i<*Empress*;i++)

{

**if** (*isCorrt*(i,a,chessboard)) {

chessboard[i][a]=1;

*Empress\_solution*(a+1, chessboard);

chessboard[i][a]=0;

}

}

}

//判断这个位置能不能放皇后

**private** **static** **boolean** isCorrt(**int** i, **int** j, **int**[][] chessboard) {

**int** s, t; //s代表行,t代表列

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

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

**return** **false**;//判断行

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

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

**return** **false**;//判断列

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

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

**return** **false**;//判断左上方

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

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

**return** **false**;//判断右下方

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

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

**return** **false**;//判断右上方

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

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

**return** **false**;//判断左下方

**return** **true**;

}

}

**四．实验结果：**

