1. 四皇后问题

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

# include <stdio.h>

# define N 4

int count=0;

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

{

int s,t;

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

if(Q[s][t]==1 && t!=j) return 0;

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

if(Q[s][t]==1 && s!=i) return 0;

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

if(Q[s][t]==1) return 0;

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

if(Q[s][t]==1) return 0;

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

if(Q[s][t]==1) return 0;

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

if(Q[s][t]==1) return 0; return 1; }

void Queue(int i, int (\*Q)[N])

{ int j,k;

if(i==N)

{ count++; printf("No.%d slove way\n",count);

for(k=0;k<N; k++)

{ for(j=0; j<N; j++) printf("%-5d", Q[k][j]);

printf("\n");}

printf("\n"); }

for(j=0;j<N; j++) { if(isCorrect(i,j,Q))

{

Q[i][j]=1; Queue(i+1,Q);

Q[i][j]=0;

}

}

}

int main()

{ int Q[N][N];

int i,j;

for (i=0; i<N; i++)

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

{ Q[i][j] = 0; printf("%-5d",Q[i][j]);

}

printf("\n");

}

printf("\n");

Queue(0, Q);

printf("solve count is %d\n", count);

getchar();

getchar();

return 0;

}

四皇后问题有两种解

我借鉴了网上的资料，但是有一个问题不是很理解。。按照这样的话8皇后要运算很久吧…

二.深度优先遍历

1.时间复杂性O（b的m次幂）

2.空间复杂度O（bm）

3.深度优先遍历优势在于空间复杂性低，可行性比广度优先更高，但是同时完整性无法保证

4.若是在深度遍历时遇到循环或者其他影响回溯都情况，则算法会出现错误。