数据结构课程设计文档

**八皇后**

小组人员：史文翰 舒星源 刘子欣

1. **问题描述**

在8X8的国际象棋上摆放8个皇后，使其不能相互攻击，即任意两个皇后都不能处于同一行，同一列或同一斜线上。

我们将问题扩充为在nXn的棋盘上摆放n个皇后，并添加了去处对称旋转可能得到的其他摆法。

1. **算法思路：**

用递归函数对每一列可能摆放的皇后位置进行尝试，并一一排除。

一个点可能在棋盘上至多存在7个相同位置的点,也就是说也要排除这七种情况.

1. **算法描述**

先请求输入棋牌大小和是否去除重复选项。递归尝试某列中可行的位置，若可行则尝试下一列，所有位置完成后与已在文件中输出的进行比对，无重复则继续输出到文件Queen.txt。所有完成后，在屏幕上打印总数。

**四、源程序**

#include<iostream>

#include<fstream>

#include<algorithm>

using namespace std;

typedef struct{

int x;

int y;

}chessman;

int whetherKill(chessman q1,chessman q2){

if(q1.y==q2.y||q1.x==q2.x)return 1;

if(q1.x-q2.x==q1.y-q2.y)return 1;

if(q1.x-q2.x==q2.y-q1.y)return 1;

return 0;

}

int compare(chessman a,chessman b){

return a.x<b.x;

}

void Transpose(chessman \*ptr,chessman \*ptr2,int n){

int i;

for(i=1;i<=n;i++){

ptr2[i].x=ptr[i].y;

ptr2[i].y=ptr[i].x;

}

sort(ptr2+1,ptr2+n+1,compare);

}

int whetherMirror(int n,chessman \*ptr,int method){

ifstream file("Queen.txt");

int i,flag=1,j=0;

char ch;

chessman \*ptr1,\*ptr2;

ptr1=new chessman[n+1];

ptr2=new chessman[n+1];

while(flag&&j<method){

for(i=1;i<=n;i++){

file>>ch>>ptr1[i].x>>ch>>ptr1[i].y>>ch;

}

file.get(ch);

Transpose(ptr,ptr2,n);

for(i=1;i<=n;i++){

if(ptr[n+1-i].y!=ptr1[i].y)break;

else if(i==n)flag=0;

}

for(i=1;flag&&i<=n;i++){

if(ptr[i].y+ptr1[i].y!=n+1)break;

else if(i==n)flag=0;

}

for(i=1;flag&&i<=n;i++){

if(ptr[n+1-i].y+ptr1[i].y!=n+1)break;

else if(i==n)flag=0;

}

for(i=1;flag&&i<=n;i++){

if(ptr1[i].y!=ptr2[i].y)break;

else if(i==n)flag=0;

}

for(i=1;flag&&i<=n;i++){

if(ptr1[n+1-i].y!=ptr2[i].y)break;

else if(i==n)flag=0;

}

for(i=1;flag&&i<=n;i++){

if(ptr1[i].y+ptr2[i].y!=n+1)break;

else if(i==n)flag=0;

}

for(i=1;flag&&i<=n;i++){

if(ptr1[n+1-i].y+ptr2[i].y!=n+1)break;

else if(i==n)flag=0;

}

j++;

}

delete []ptr1;

delete []ptr2;

file.close();

return flag;

}

void outputQ(chessman \*ptr,int n,ofstream &file){

int i;

for(i=1;i<=n;i++){

file<<"("<<ptr[i].x<<","<<ptr[i].y<<")";

}

file<<endl;

}

void Queen(int n,int row,chessman \*ptr,int &method,int mirror,ofstream &file){

if(row==n+1){

if(mirror||whetherMirror(n,ptr,method)){

outputQ(ptr,n,file);

method++;

}

}

else{

int i,j,flag=1;

for(i=1;i<=n;i++){

chessman Q;

flag=1;

Q.x=row,Q.y=i;

for(j=1;j<row;j++){

if(whetherKill(ptr[j],Q))flag=0;

}

if(flag){

ptr[row]=Q;

Queen(n,row+1,ptr,method,mirror,file);

}

}

}

}

int main(){

int n,method=0,mirror;

cout<<"Please input the size of chessboard:";

cin>>n;

cout<<"Want to have the mirror example? 1---yes 0---no"<<endl;

cin>>mirror;

if(mirror)mirror=1;

chessman \*ptr;

ptr=new chessman[n+1];

ofstream file("Queen.txt");

Queen(n,1,ptr,method,mirror,file);

cout<<"There is "<<method<<" methods."<<endl;

cout<<"The answer is in file<Queen>."<<endl;

delete []ptr;

file.close();

return 0;

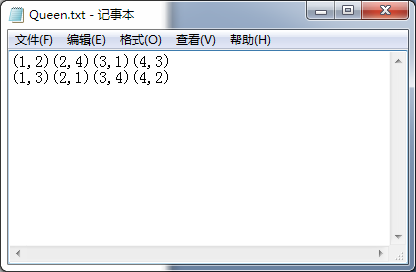
}

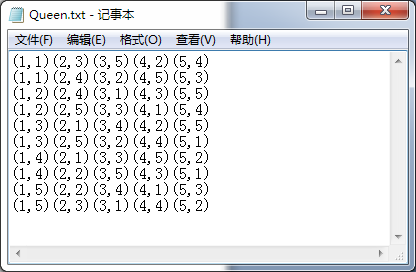
**五、测试数据**

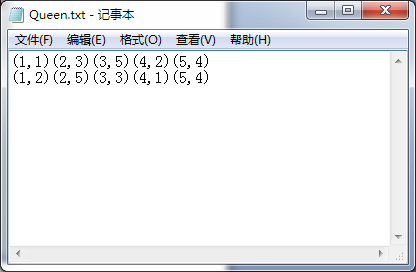
1:(1,1)

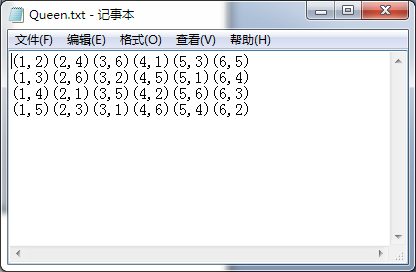
2:0

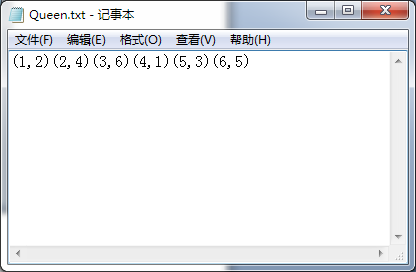
3:0

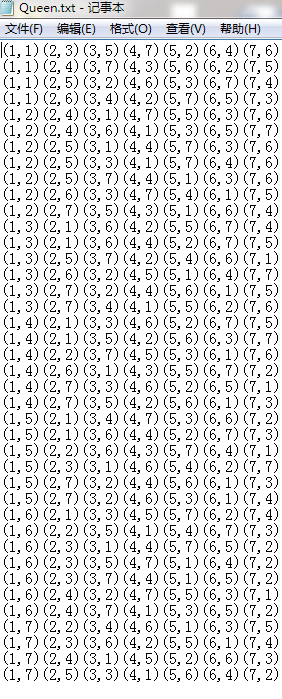
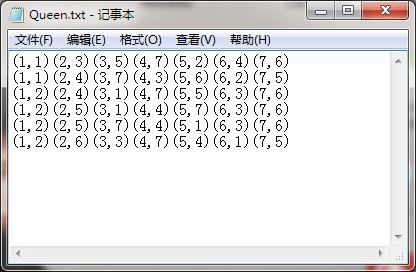
4: 两个互相对称解

5: -----1

----0

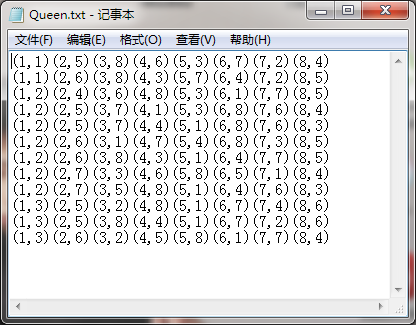
6: ----1

----0

7: ---140种/6种

8:92种

12种



**六、结果分析和结论**

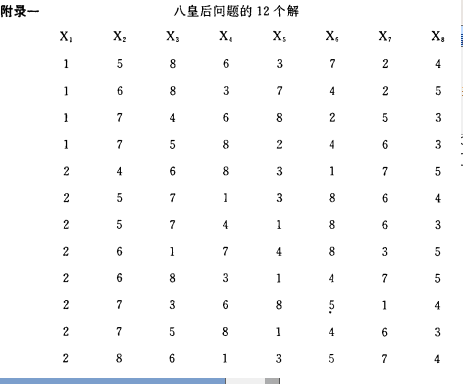
程序运行正常,在成功排列出92种的基础上,我们估算的12种无重复解也得到了证实,算法无误.

**七、心得体会**

史文翰：在测试方面，由于这个题是单输入多输出问题，输入测试变得异常简单，但 在解决大数字皇后问题的时候，计算机无法在规定时间内给出确定解，这也就提醒了测 试者，如何取测试数字是合理的，且对于一些容错（比如输入某个字符而不是数字）， 也应该有相应的测试处理。

舒星源：在修改排除重复时遇到了很多问题.首先将棋盘与旋转后的比对,相当于与转置后的位置矩阵比对,虽然老师已经教过快速转置的方法,但这个方法在每一行只有一个值时没有明显优越性,所以简单的使用了老方法.在校验最后12个解时,我们参考了知网上的论文,然后发现,论文中的解是有重复有遗漏的.

刘子欣：写文档的话也是一个锻炼能力的活儿，首先必须要能够读懂别人写的代码，而且总结方面的工作也是很考验一个人的能力的。



举例， 2 8 6 1 3 5 7 4

转置后为(1,4)(2,1)(3,5)(4,8)(5,6)(6,3)(7,7)(8,2)

与 (1,2)(2,7)(3,3)(4,6)(5,8)(6,5)(7,1)(8,4)对称

即顺时针旋转90度与他的第六组重合。

面对结果难以验算的情况，有时候连网上的成例都不能相信，更多时候还是要自己研究思考。

**八、分工及签名**

编码：舒星源 测试：史文翰 文档:刘子欣