N皇后问题

1.项目简介

> 八皇后问题是一个古老而著名的问题，是回溯算法的经典问题。该问题是十九世纪著名的数学家高斯在1850年提出的：在8*8的国际象棋棋盘上，安放8个皇后，要求没有一个皇后能够“吃掉”任何其它一个皇后，即任意两个皇后不能处于同一行，同一列或者同一条对角线上，求解有多少种摆法。 高斯认为有76种方案。1854年在柏林的象棋杂志上不同的作者发表了40种不同的解，后来有人用图论的方法得到结论，有92中摆法。本实验拓展了N皇后问题，即皇后个数由用户输入

2.项目功能要求

八皇后在棋盘上分布的各种可能的格局数目非常大，约等于2的32次方种，但是，可以将一些明显不满足问题要求的格局排除掉。由于任意两个皇后不能同行，即每行只能放置一个皇后，因此将第i个皇后放在第i航上，这样在放置第i个皇后时，只要考虑它与前i-1个皇后处于不同列和不同对角线位置上即可。
解决这个问题采用回溯法，首先将第一个皇后放置在第一行第一列，然后，依次在下一行上放置一个皇后，直到八个皇后全部放置安全。在放置每个皇后时，都依次兑每一列进行检测，首先检测放在第一列是否与已放置的皇后冲突，如不冲突，则将皇后放置在该列，否则，选择改行的下一列进行检测。如整行的八列都冲突，则回到上一行，重新选择位置，依次类推。

3.n皇后算法思想

n皇后问题是一个典型的dfs问题，大体思路就是从第一个列填到第n列，对于每一列填入的行数进行检查，如果符合要求就接着填入下一个位置，如果不符合就返回。 这里我采用了一种简单的方式优化了n皇后的求解问题，就是开辟一个一维数组，用数组的下标代表列号，数组内储存该列放入皇后的行数。同时也就避免了填入的在同一列上的问题。这也就是为什么我只开辟了一个map的一维数组。

4.核心代码及功能

- **新建的变量**

```
int Count=0;                //n皇后问题解的个数
int totalNum;               //n的值（即皇后的个数）
int map[100];               //这里默认最大n为99，下标从1开始使用（即99*99）
```

- **dfs函数**
  pos参数为搜索到了第pos列，每次填入后pos+1，调用下一个dfs函数，当pos>n（totalNum）时返回输出当前解。这里抽象出一个Try函数用来判断是否可以填入第i行，第pos列。Try函数实现见下一个函数

```
void dfs(int pos){          //深搜寻找解
    if(pos>totalNum){       //当最后一列填完后就输出解
        Count++;
        display();
    }
    else{                   //从1-totalNum试可否放入第pos列第i行
        for(int i=1;i<=totalNum;i++){
            if(Try(pos,i)){         //如果可行就填入并进入下一个位置
                map[pos]=i;
                dfs(pos+1);
            }
        }
    }
}
```

- **Try函数**
  参数x代表填入的列数，y代表试的行数，首先根据题意不能和之前的在同一列（这个已经根据我的开辟数组解决了）其次不能在同一行，所以有pos之前填入的所有数值必须满足`map[j]!=y`，最后不能在同一条对角线上，所以需要之前的填入的棋子满足`abs(x-j)!=abs(map[j]-y`

```
bool Try(int x,int y){       //检查该摆放是否可行
    int j=1;
    while(j<x){              //可行条件为之前放入的行数和y不重复
                             //同时不能和之前放入的在对角线上
        if(map[j]==y||(abs(x-j)==abs(map[j]-y))){
            return false;
        }
        j++;
    }
    return true;
}
```

- **display函数**

```
void display(){              //展示可行解的摆放方式
    for(int i=1;i<=totalNum;i++){
        for(int j=1;j<=totalNum;j++){
            if(map[i]==j){
                cout<<'X'<<' ';
            }
            else{
                cout<<'0'<<' ';
            }
        }
        cout<<endl;
    }
    cout<<endl;
}
```

5.项目演示

```
/Users/kirito/CLionProjects/untitled/cmake-build-debug/untitled
请输入皇后的个数:5
皇后摆法:
X 0 0 0 0
0 0 X 0 0
0 0 0 0 X
0 X 0 0 0
0 0 0 X 0

X 0 0 0 0
0 0 0 X 0
0 X 0 0 0
0 0 0 0 X
0 0 X 0 0

0 X 0 0 0
0 0 0 X 0
X 0 0 0 0
0 0 X 0 0
0 0 0 0 X

0 X 0 0 0
0 0 0 0 X
0 0 X 0 0
X 0 0 0 0
0 0 0 X 0
```

```
0 0 X 0 0
X 0 0 0 0
0 0 0 X 0
0 X 0 0 0
0 0 0 0 X

0 0 X 0 0
0 0 0 0 X
0 X 0 0 0
0 0 0 X 0
X 0 0 0 0

0 0 0 X 0
X 0 0 0 0
0 0 X 0 0
0 0 0 0 X
0 X 0 0 0

0 0 0 X 0
0 X 0 0 0
0 0 0 0 X
0 0 X 0 0
X 0 0 0 0

0 0 0 0 X
0 X 0 0 0
0 0 0 X 0
X 0 0 0 0
0 0 X 0 0

0 0 0 0 X
0 0 X 0 0
X 0 0 0 0
0 0 0 X 0
0 X 0 0 0

共有10种解法

Process finished with exit code 0
```