回溯算法解决八皇后问题

171491103 米日阿地·买买提明 硬件一班

**一实验题目：**

回溯搜索算法

**二.算法简介**

回溯法（探索与回溯法）是一种选优搜索法，又称为试探法，按选优条件向前搜索，以达到目标。但当探索到某一步时，发现原先选择并不优或达不到目标，就退回一步重新选择，这种走不通就退回再走的技术为回溯法，而满足回溯条件的某个状态的点称为“回溯点”。

。

三．**八皇后问题简介**

在国际象棋中，皇后是最强大的一枚棋子，可以吃掉与其在同一行、列和斜线的敌方棋子。比中国象棋里的车强几百倍，比她那没用的老公更是强的飞起（国王只能前后左右斜线走一格）。  
八皇后问题是这样一个问题：将八个皇后摆在一张8\*8的国际象棋棋盘上，使每个皇后都无法吃掉别的皇后，一共有多少种摆法？  
八皇后问题，是一个古老而著名的问题，是回溯算法的典型案例。该问题是国际西洋棋棋手马克斯·贝瑟尔于1848年提出。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解出92种结果。计算机发明后，有多种计算机语言可以解决此问题。

**三：实验代码**

**#include<iostream>**

**#include<cmath>**

**using namespace std;**

**/\***

**1. 用一个一维数组来表示八皇后的摆放情况**

**x[i]表示第i个皇后放在第i行的第x[i]列**

**2.剪枝条件**

**不同行 每一行放一个皇后**

**不同列 x[i] != x[j]**

**不同斜线 |i - j| != |x[i] - x[j]| 即 abs(i - j) != abs(x[i] - x[j])**

**\*/**

**int num;**

**int \*x;**

**int sum;**

**bool check(int k){ //剪枝**

**for(int j=1;j<k;j++){**

**if( abs(k - j) == abs(x[k] - x[j]) || x[j] == x[k] )**

**return false;**

**}**

**return true;**

**}**

**void backtrack(int t){**

**if(t > num){**

**sum++;**

**for(int m=1; m<=num; m++){**

**cout<<"<"<<m<<","<<x[m]<<">";**

**}**

**cout<<endl;**

**}**

**else{**

**for(int i=1; i<=num; i++){**

**x[t] = i;**

**if( check(t) ){**

**backtrack(t+1);**

**}**

**}**

**}**

**}**

**int main(){**

**num = 8;**

**sum = 0;**

**x = new int [num+1];**

**for(int i=0; i<=num; i++){**

**x[i] = 0;**

**}**

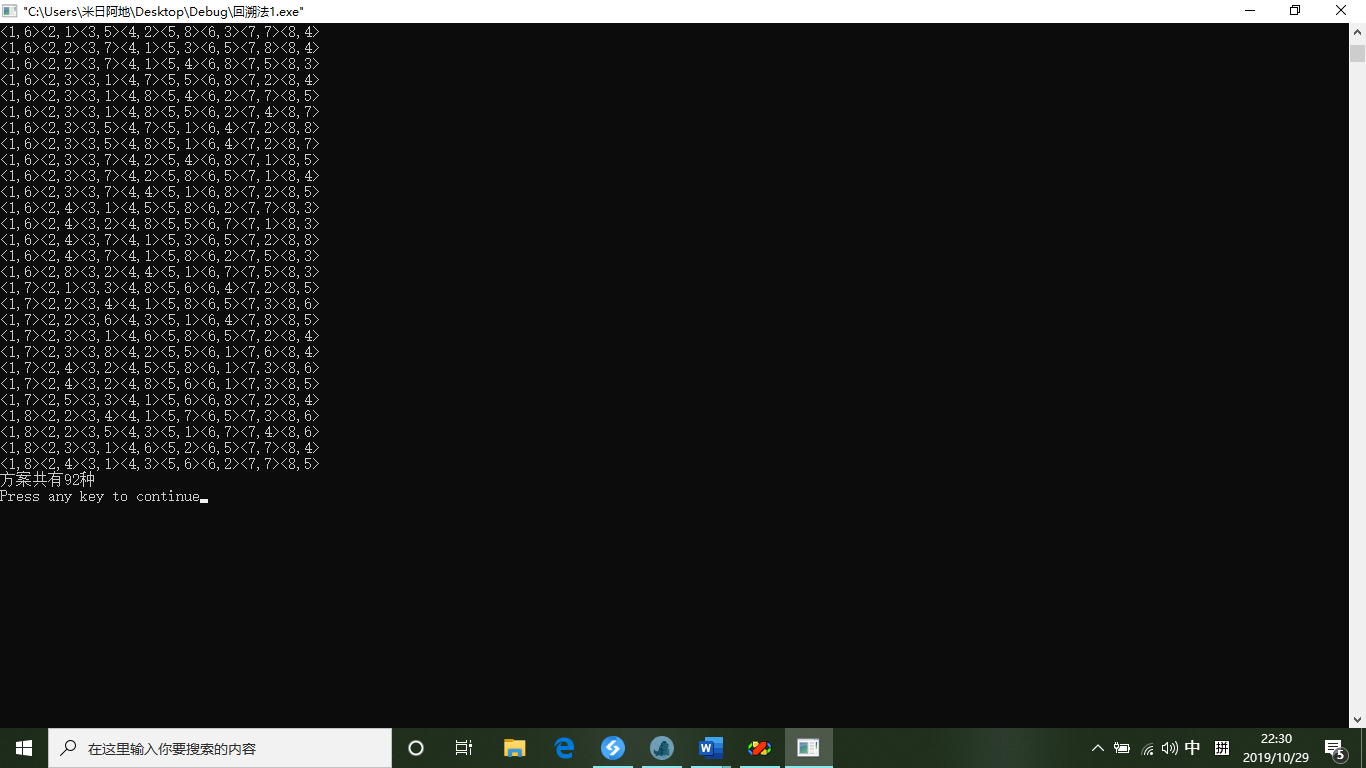
**backtrack(1);**

**cout<<"方案共有"<<sum<<"种"<<endl;**

**delete []x;**

**}**

**四：实验截图**



**五：实验总结**

分析主要代码算法是逐行安排皇后的，其参数row为现在正执行到第几行。n是皇后数，在八皇后问题里当然就是8啦。  
if(row == n)这句代码好理解，如果程序执行了row == n，说明从0到n-1的位置都放上了皇后，那自然是找到了一种解法，于是八皇后问题解法数加1。  
否则进入else语句。遍历所有列col，将当前col存储在数组c里，然后使用check()检查row行col列能不能摆皇后，若能摆皇后，则递归调用queen去安排下一列摆皇后的问题。

还不太清楚？再慢点来，刚开始的时候row = 0，意思是要对第0行摆皇后了。  
If判断失败，进入else，进入for循环，col初始化为0  
显然，0行0列的位置一定可以摆皇后的，因为这是第一个皇后啊，后宫空荡她想怎么折腾就怎么折腾，于是check(0)测试成功，递归调用queen(1)安排第1行的皇后问题。

皇后放在第1行时即row=1，进来if依然测试失败，进入for循环，col初始化为0。1行0列显然是不能摆皇后的，因为0行0列已经有一个圣母皇太后在那搁着了，于是check()测试失败，循环什么也不做空转一圈，col变为1。1行1列依然check()测试失败，一直到1行2列，发现可以摆皇后，于是继续递归queen(2)去安排第二个皇后位置。

如果在某种情况下问题无解呢？例如前面在4皇后问题中，0行0列摆皇后是无解的。假设前面递归到queen(2)时候，发现第2行没有地方可以摆皇后，那怎么办呢？要注意queen(2)的调用是在queen(1)的for循环框架内的，queen(2)若无解，则自然而然queen(1)的for循环col自加1，即将第1行的皇后从1行2列改为1行3列的位置，检查可否放皇后后继续安排下一行的皇后。如此递归，当queen(0)的col自加到n-1，说明第一列的皇后已经遍历了从0行1列到0行n-1列，此时for循环结束，程序退出。

在主函数中调用queen(0)，得到正确结果，8皇后问题一共有92种解法