# 解题报告模板

**问题名称和编号**

**1 问题描述**

在N\*N的方格棋盘放置了N个皇后，使得它们不相互攻击（即任意2个皇后不允许处在同一排，同一列，也不允许处在与棋盘边框成45角的斜线上。  
你的任务是，对于给定的N，求出有多少种合法的放置方法。

**2 输入**

共有若干行，每行一个正整数N≤10，表示棋盘和皇后的数量；如果N=0，表示结束。

**3 输出**

共有若干行，每行一个正整数，表示对应输入行的皇后的不同放置数量。

**4 样例输入**

1

8

5

0

**5 样例输出**

1

92

10

[**http://acm.hdu.edu.cn/showproblem.php?pid=2553**](http://acm.hdu.edu.cn/showproblem.php?pid=2553) **题目地址**

**6 问题分析**

利用回溯法，从第一行第一列开始放棋子开始，慢慢的尝试n行放n个棋子的可行性，一旦其中一个分支无法进行下去，则立即返回到原来的位置继续开始尝试，尝试出一种解法时，则将变量cnt+1，然后接着尝试，直到尝试完所有的可能的摆法，就结束程序。然后这边为了让acm不超时，用了一个数组cun[]来存储已经走过的n个棋子的解法的数量。

**7 程序代码**

#include<stdio.h>

#include<math.h>

int rank[20],cun[20];

bool vis[20],hao[20];

int n,cnt=0;

void dfs(int pos){

if(hao[n]==1){

cnt=cun[n];

return;

}

if(n!=0&&pos==n+1){//递归边界条件

cnt++;

return;

}

for(int i=1;i<=n;i++){//枚举每行

if(vis[i]==false){

bool flag=true;

for(int j=1;j<pos;j++){//枚举pos之前的皇后

if(abs(pos-j)==abs(i-rank[j])){

flag=false;

break;

}

}

if(flag){

rank[pos]=i;//pos列在i行

vis[i]=true;

dfs(pos+1);

vis[i]=false;

}

}

}

}

void bbq(){

scanf("%d",&n);

dfs(1);

if(n!=0){

hao[n]=1;

cun[n]=cnt;

printf("%d\n",cun[n]);

cnt=0;

bbq();

}

}

int main(){

bbq();

}

**8 测试示例**

**8**

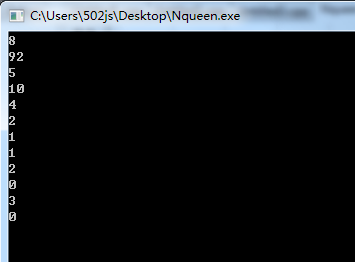
**5**

**4**

**1**

**2**

**3**



**9 结果截图**





**10 解题备注**

其实在本地运行完全问题不大，主要是面对acm，如果你程序解题时间过长，他会自动判断为超时，老师说因为acm后台是在循环给出数字来测试代码，所以要多建立一个数组来存储已经输出过的值，避免重复递归消耗时间。