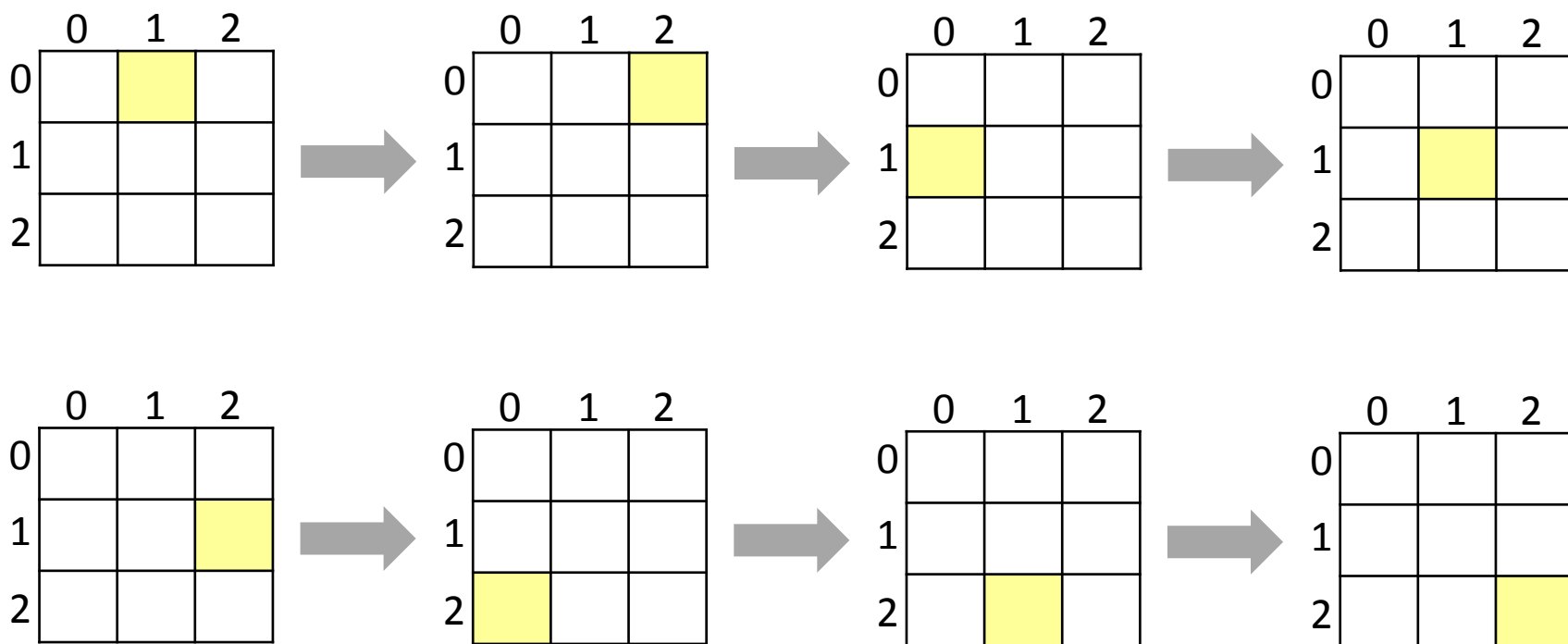


# C++算法

# 棋盘格下一格

参照以下图片，在 $n \times n$ 格棋盘上，行号编号0到 $n-1$ ，列号编号0到 $n-1$ 。有一个棋子在 $x$ 号行 $y$ 号列，请输出该棋子的下一格在几号行几号列。



```

4  int n,x,y,nx,ny;
5  cin>>n>>x>>y;
6  if(y==n-1){
7      ny=0;
8      nx=x+1;
9  }
10 else{
11     ny=y+1;
12     nx=x;
13 }
14 cout<<nx<<" "<<ny<<endl;

```



```

nx=(y==n-1?x+1:x);
ny=(y==n-1?0:y+1);

```

	0	1	2
0			
1			
2			



	0	1	2
0			
1			
2			



	0	1	2
0			
1			
2			



	0	1	2
0			
1			
2			

# 深度优先搜索

depth-first search

## 棋盘格独立集问题

# 独立集

安排**互不冲突**的个体

独立

```
graph BT; A[独立] --> B[安排互不冲突的个体]
```

The diagram consists of two rectangular boxes. The top box is light gray and contains the text '安排互不冲突的个体' (Arranging non-conflicting individuals). The bottom box is light orange and contains the text '独立' (Independent). A brown arrow points vertically from the top of the bottom box to the bottom of the top box, indicating a relationship or flow from 'Independent' to 'Arranging non-conflicting individuals'.

# 最大独立集

尽量多地安排互不冲突的个体

最大

独立

# 独立集计数

安排互不冲突的个体有几种方案

独立

计数

# 四个斜眼枪手

在3\*3格棋盘上要放四个斜眼枪手。每个斜眼枪手可以朝四个斜方向开枪：左上/右上/左下/右下。要求枪手互相不在对方开枪方向上，请输出所有摆放方案，1代表枪手0代表空位。

111
000
010

110
000
110

101
101
000

001
101
001

如何枚举  
所有方案



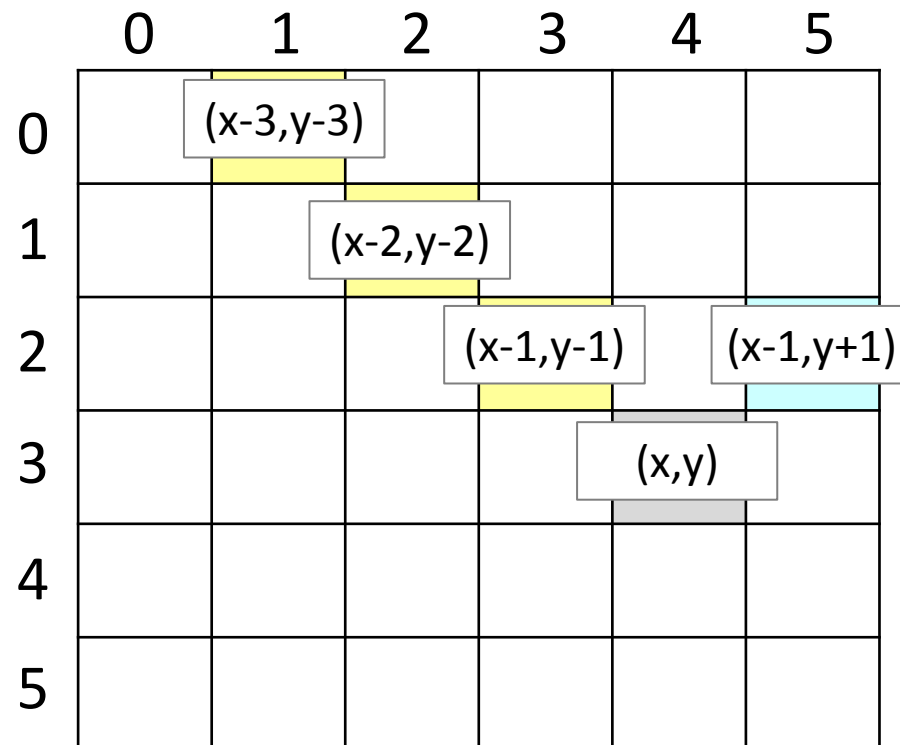
$f[a][b]$ 表示(a,b)是否已有枪手

判断(x,y)格能否放新枪手

```
12 bool valid(int x,int y){
13     for(int i=1;i<=min(x,y);i++)
14         if(f[x-i][y-i])return 0;
15     for(int i=1;i<=min(x,N-1-y);i++)
16         if(f[x-i][y+i])return 0;
17     return 1;
18 }
```

检查左上方

检查右上方



```
3 const int N=3;  
4 const int GUNS=4;
```

从(x,y)格子继续枚举,已经放了c个枪手

```
19 void dfs(int x,int y,int c){  
20     if(c==GUNS){ans++;print();return;}  
21     if(x==N)return;  
22     int nx=(y==N-1?x+1:x);  
23     int ny=(y==N-1?0:y+1);  
24     if(valid(x,y)){  
25         f[x][y]=1;  
26         dfs(nx,ny,c+1);  
27         f[x][y]=0;  
28     }  
29     dfs(nx,ny,c);  
30 }  
31 int main(){  
32     dfs(0,0,0);  
33     return 0;  
34 }
```

计算下一格  
行号nx和列号ny

判断(x,y)格能否放新枪手

从(nx,ny)格子继续枚举  
已经放了c+1个枪手

从(nx,ny)格子继续枚举  
已经放了c个枪手

f[x][y]表示(x,y)是否已有枪手

```
3 const int N=3;  
4 const int GUNS=4;
```

从(x,y)格子继续枚举,已经放了c个枪手

```
19 void dfs(int x,int y,int c){  
20     if(c==GUNS){ans++;print();return;}  
21     if(x==N)return;  
22     int nx=(y==N-1?x+1:x);  
23     int ny=(y==N-1?0:y+1);  
24     if(valid(x,y)){  
25         f[x][y]=1;           ← 摆放  
26         dfs(nx,ny,c+1);  
27         f[x][y]=0;           ← 撤销  
28     }  
29     dfs(nx,ny,c);  
30 }  
31 int main(){  
32     dfs(0,0,0);  
33     return 0;  
34 }
```

摆放

撤销

对照  
语句

# 六个斜眼枪手

在4\*4格棋盘上要放六个斜眼枪手。每个斜眼枪手可以朝四个斜方向开枪：左上/右上/左下/右下。要求枪手互相不在对方开枪方向上，请输出所有摆放方案，1代表枪手0代表空位。

1	1	1	1
0	0	0	0
0	0	0	0
0	1	1	0

1	1	1	0
0	0	0	0
0	0	0	0
1	1	1	0

1	1	0	1
0	0	0	1
1	0	0	0
0	0	1	0

1	1	0	0
0	0	0	1
1	0	0	0
1	0	1	0

如何枚举  
所有方案

```
3 const int N=4;
4 const int GUNS=6;
```

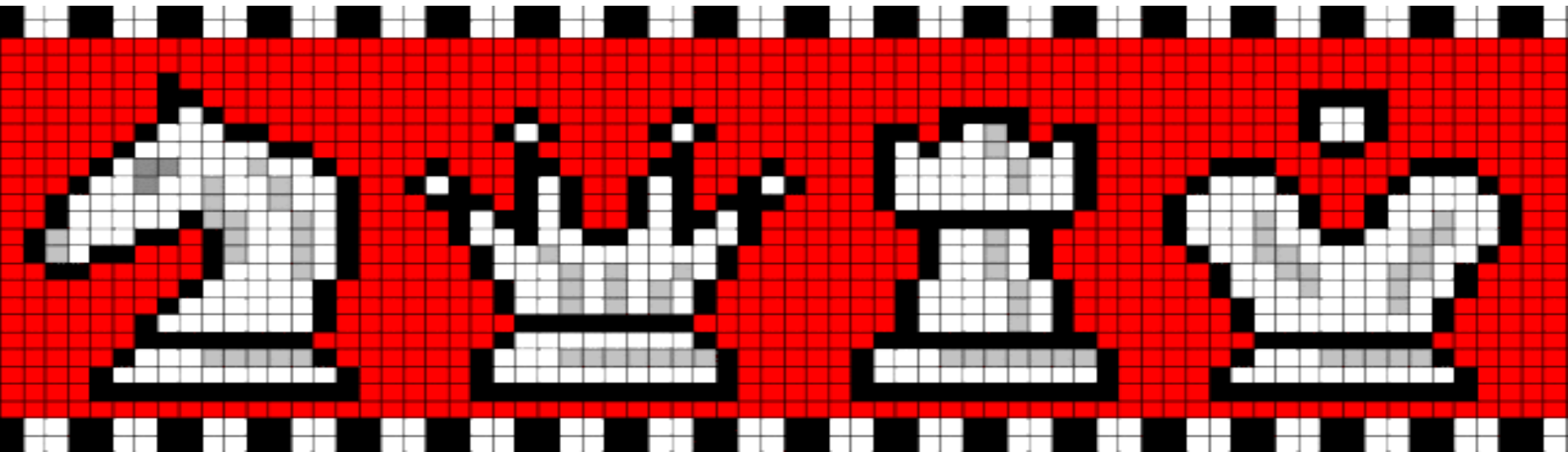
```
19 void dfs(int x,int y,int c){
20     if(c==GUNS){ans++;print();return;}
21     if(x==N)return;
22     int nx=(y==N-1?x+1:x);
23     int ny=(y==N-1?0:y+1);
24     if(valid(x,y)){
25         f[x][y]=1;
26         dfs(nx,ny,c+1);
27         f[x][y]=0;
28     }
29     dfs(nx,ny,c);
30 }
31 int main(){
32     dfs(0,0,0);
33     return 0;
34 }
```

跟着老师翻译  
理解每一行

# 八皇后

著名的八皇后问题是国际象棋棋手贝瑟尔于1848年提出

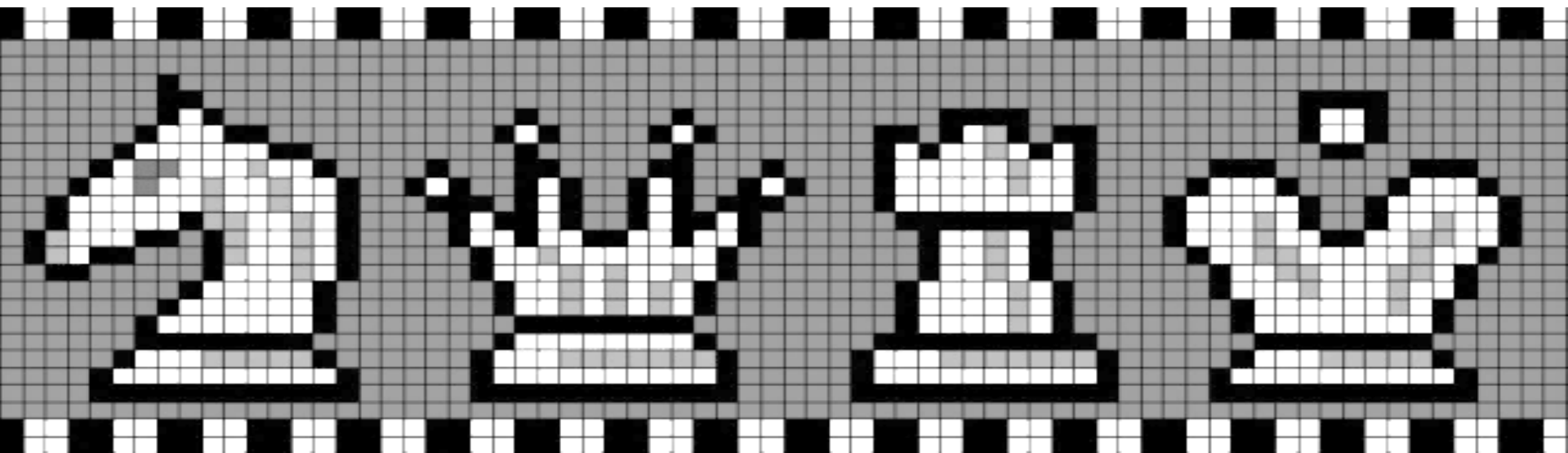
在 $8 \times 8$ 格的国际象棋上摆放八个皇后，使其不能互相攻击，即任意两个皇后都不能处于同一行、同一列或同一斜线上，问有多少种摆法



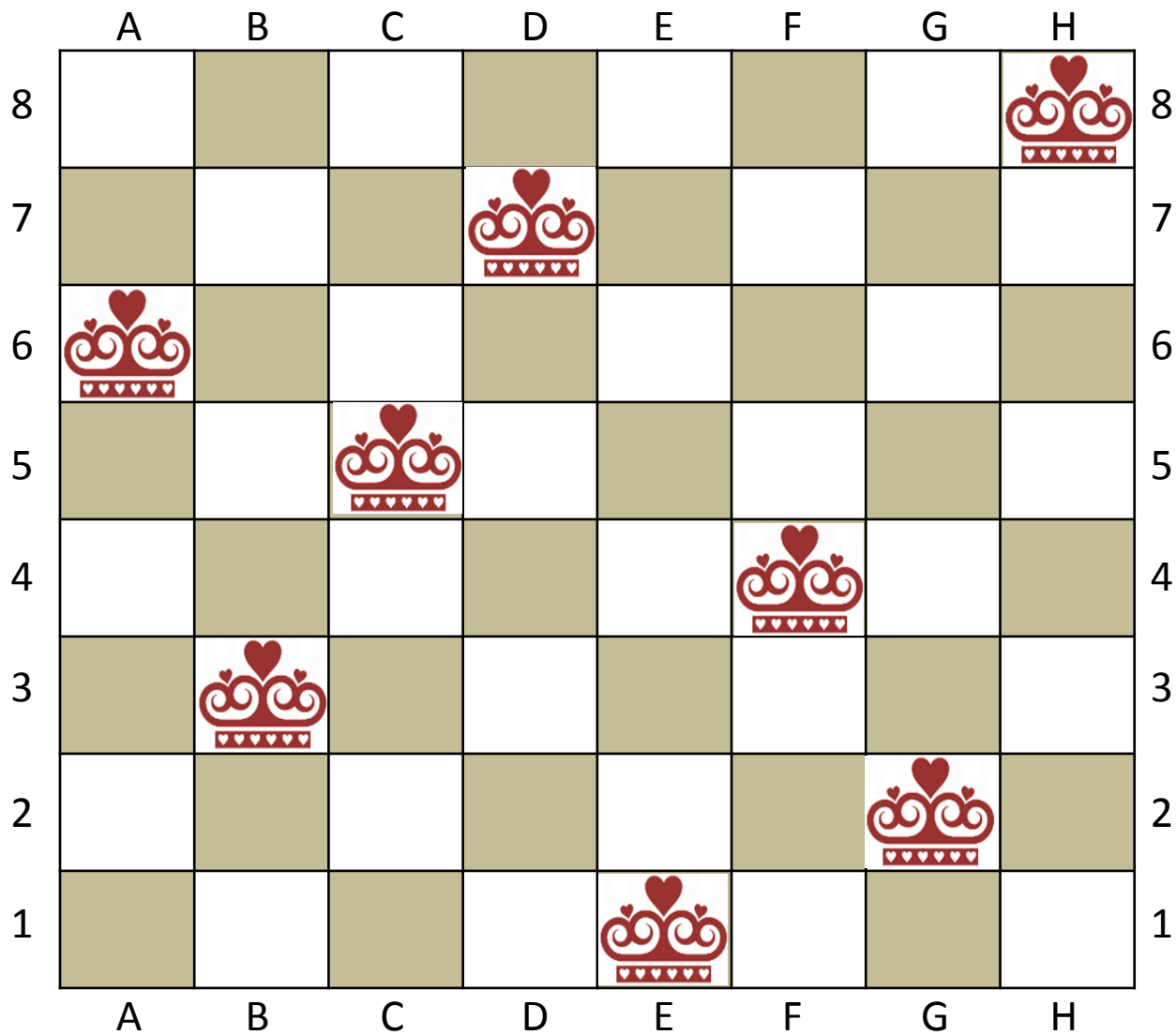
# 简化问题：四皇后

用纸和笔枚举所有四皇后摆放方案

在 $4 \times 4$ 格的国际象棋上摆放4个皇后，使其不能互相攻击，即任意两个皇后都不能处于同一行、同一列或同一斜线上，问有多少种摆法



共有多少  
种摆法





# 八皇后：方案计数

共 $8*8=64$ 格

每一格可以放或者不放皇后

暴力枚举：共 $2^{64}$ 种可能性  
判断皇后是否冲突

# 八皇后：独立集计数问题

优化  
枚举

选择放置皇后时  
确保没有冲突

算法1

逐格  
放置

对每一格  
考虑是否放皇后

是非题

算法2

逐行  
放置

对每一行  
考虑放皇后在哪一列

选择题






# 逐格放置

```
22 void dfs(int x,int y,int c){
23     if(c==N){ans++;print();return;}
24     if(x==N)return;
25     int nx=(y==N-1?x+1:x);
26     int ny=(y==N-1?0:y+1);
27     if(valid(x,y)){
28         f[x][y]=1;
29         dfs(nx,ny,c+1);
30         f[x][y]=0;
31     }
32     dfs(nx,ny,c);
33 }
34 int main(){
35     dfs(0,0,0);
36     return 0;
37 }
```

```
11 bool valid(int x,int y){  
12     for(int i=1;i<=min(x,y);i++)  
13         if(f[x-i][y-i])return 0;  
14     for(int i=1;i<=min(x,N-1-y);i++)  
15         if(f[x-i][y+i])return 0;  
16     for(int i=0;i<x;i++)  
17         if(f[i][y])return 0;  
18     for(int i=0;i<y;i++)  
19         if(f[x][i])return 0;  
20     return 1;  
21 }
```

判断(x,y)格能否放新皇后

```
12 bool valid(int x,int y){  
13     return !clmn[y]&&!d1[x+y]&&!d2[x-y+N-1];  
14 }
```






	0	1	2	3	4	5	6	7
0								
1								
2								
3								
4								
5								
6								
7								

clmn[y]表示y号列  
是否已被占领

y	0	1	2	3	4	5	6	7
clmn[y]	0	1	1	1	0	1	0	1

判断(x,y)格能否放新皇后

```
12 bool valid(int x,int y){  
13     return !clmn[y]&&!d1[x+y]&&!d2[x-y+N-1];  
14 }
```

	0	1	2	3	4	5	6	7	
									8
									9
									10
									11
									12
									13
									14






斜线编号0到14  
共15条

d1[k]表示k号正斜线  
是否已被占领

k	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
d1[k]	0	1	0	0	1	0	1	1	0	0	1	0	0	0	0

判断(x,y)格能否放新皇后

```
12 bool valid(int x,int y){  
13     return !clmn[y]&&!d1[x+y]&&!d2[x-y+N-1];  
14 }
```

	7	6	5	4	3	2	1	0
8								
9								
10								
11								
12								
13								
14								

斜线编号0到14  
共15条

d2[k]表示k号反斜线  
是否已被占领

k	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
d2[k]	0	0	0	1	1	1	1	0	0	1	0	0	0	0	0

# 逐行放置

```
12 bool valid(int x,int y){
13     return !clmn[y]&&!d1[x+y]&&!d2[x-y+N-1];
14 }
15 void dfs(int x){
16     if(x==N){ans++;print();return;}
17     for(int y=0;y<N;y++)
18         if(valid(x,y)){
19             f[x][y]=clmn[y]=d1[x+y]=d2[x-y+N-1]=1;
20             dfs(x+1);
21             f[x][y]=clmn[y]=d1[x+y]=d2[x-y+N-1]=0;
22         }
23 }
24 int main(){
25     dfs(0);
26     return 0;
27 }
```



快快编程520

快快编程  
kkcoding.net

```
19 int main(){
20     for(int i=0;i<N;i++){
21         for(int j=0;j<N;j++){
22             char ch;
23             cin>>ch;
24             if(ch=='Q'){
25                 row[i]=1;
26                 clmn[j]=1;
27             }
28         }
29     }
30     dfs(0);
31     cout<<ans<<endl;
32     return 0;
33 }
34 }
```



快快编程  
kkcoding.net

```
9 void dfs(int x){
10     if(x==N){ }
11     if(row[x]){ }
12     for(int y=0;y<N;y++)
13         if(valid(x,y)){
14             clmn[y]=d1[x+y]=d2[x-y+N-1]=1;
15             dfs(x+1);
16         }
17 }
18 }
```

```
6 bool valid(int x,int y){  
7     return   
8 }
```

快快编程521

快快编程  
kkcoding.net

```
6 bool valid(int x,int y){  
7     return !clmn[y]&& d[x][y]=='o';  
8 }
```

请同学翻译第七行

从第x行继续枚举，已经放了num个

```
9 void dfs(int x, int num){
10     if(num==k) { }
11     if(x==n) return;
12     for(int y=0; y<n; y++){
13         if(valid(x, y)){
14             clmn[y]=1;
15             dfs(x+1, num+1);
16             clmn[y]=0;
17         }
18     }
19 }
```

第x行不放了  
从第x+1行开始继续枚举

# 快快编程作业

520

521

拓展题

522