



实验舱
青少年编程
走近科学 走进名校

挑战信息学奥林匹克

C++教程 (302)
二维数组 (2)

二维数组的存储

- 逻辑存储
- 物理存储

问题：

数组元素 $a[2][3]$ 在内存的
什么位置？

$[0][0]$	$[0][1]$	$[0][2]$	$[0][3]$	$[0][4]$	$[0][5]$
$[1][0]$	$[1][1]$	$[1][2]$	$[1][3]$	$[1][4]$	$[1][5]$
$[2][0]$	$[2][1]$	$[2][2]$	$[2][3]$	$[2][4]$	$[2][5]$
$[3][0]$	$[3][1]$	$[3][2]$	$[3][3]$	$[3][4]$	$[3][5]$

a	$a+1$	$a+2$	$a+23$

例1：同行列对角线的格子

描述

输入三个自然数N，i，j（ $1 \leq i \leq N$ ， $1 \leq j \leq N$ ），输出在一个N*N格的棋盘（行列均从1开始编号），与格子（i，j）同行、同列、同一对角线的所有格子的位置。

如：n=4，i=2，j=3表示了棋盘中的第二行第三列的格子，如下图：

当n=4，i=2，j=3时，输出的结果是：

- (2,1) (2,2) (2,3) (2,4) 同一行上格子的位置
- (1,3) (2,3) (3,3) (4,3) 同一列上格子的位置
- (1,2) (2,3) (3,4) 左上到右下对角线上的格子的位置
- (4,1) (3,2) (2,3) (1,4) 左下到右上对角线上的格子的位置

第一列	第二列	第三列	第四列	
				第一行
		(2,3)		第二行
				第三行
				第四行

例1：同行列对角线的格子

样例输入

4 2 3

样例输出

(2,1) (2,2) (2,3) (2,4)

(1,3) (2,3) (3,3) (4,3)

(1,2) (2,3) (3,4)

(4,1) (3,2) (2,3) (1,4)

第一列	第二列	第三列	第四列	
				第一行
		(2,3)		第二行
				第三行
				第四行

输出行

(1, 1)	(1, 2)	(1, 3)	(1, 4)
(2, 1)	(2, 2)	(2, 3)	(2, 4)
(3, 1)	(3, 2)	(3, 3)	(3, 4)
(4, 1)	(4, 2)	(4, 3)	(4, 4)

```
for(int i=1;i<=n;i++)  
{  
    cout<<"("<<x<<","<<i<<")"<<" ";  
}  
cout<<endl;
```

输出列

(1, 1)	(1, 2)	(1, 3)	(1, 4)
(2, 1)	(2, 2)	(2, 3)	(2, 4)
(3, 1)	(3, 2)	(3, 3)	(3, 4)
(4, 1)	(4, 2)	(4, 3)	(4, 4)

```
for(int i=1;i<=n;i++)  
{  
    cout<<"("<<i<<","<<y<<")"<<" ";  
}  
cout<<endl;
```

输出左上到右下对角线

(1, 1)	(1, 2)	(1, 3)	(1, 4)
(2, 1)	(2, 2)	(2, 3)	(2, 4)
(3, 1)	(3, 2)	(3, 3)	(3, 4)
(4, 1)	(4, 2)	(4, 3)	(4, 4)

对角线上数组元素下标的关系:

$$y - x = j - i$$

```
for(int i=1;i<=n;i++)
{
    for(int j=1;j<=n;j++)
    {
        if((j-i)==(y-x))
            cout<<"("<<i<<","<<j<<")"<<" ";
    }
}
cout<<endl;
```

输出左下到右上对角线

(1, 1)	(1, 2)	(1, 3)	(1, 4)
(2, 1)	(2, 2)	(2, 3)	(2, 4)
(3, 1)	(3, 2)	(3, 3)	(3, 4)
(4, 1)	(4, 2)	(4, 3)	(4, 4)

对角线上数组元素下标的关系:

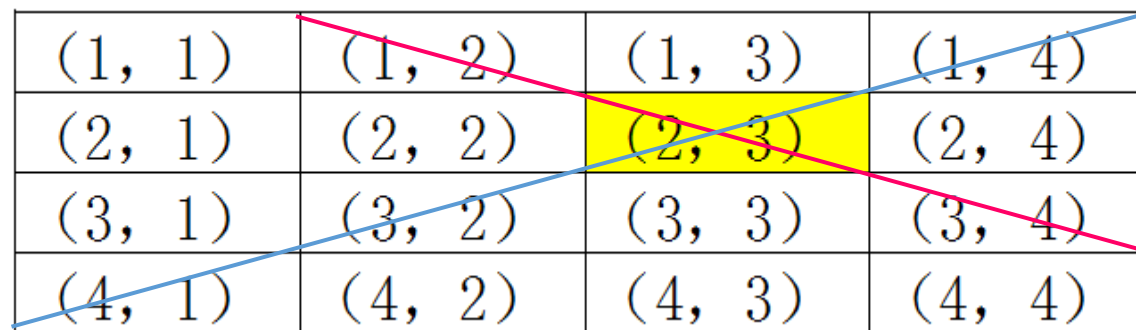
$$x + y = i + j$$

```
for(int i=n;i>=1;i--)  
{  
    for(int j=n;j>=1;j--)  
    {  
        if((i+j)==(y+x))  
            cout<<"("<<i<<","<<j<<")"<<" ";  
    }  
}  
cout<<endl;
```


思考：能否直接输出对角线上的数据？

输出左上到右下对角线： $y - x = j - i$

输出左下到右上对角线： $x + y = i + j$



(1, 1)	(1, 2)	(1, 3)	(1, 4)
(2, 1)	(2, 2)	(2, 3)	(2, 4)
(3, 1)	(3, 2)	(3, 3)	(3, 4)
(4, 1)	(4, 2)	(4, 3)	(4, 4)

思考：能否直接输出对角线上的数据？

输出左上到右下对角线：

$i, y - x + i$

输出左下到右上对角线：

$i, x + y - i$

(1, 1)	(1, 2)	(1, 3)	(1, 4)
(2, 1)	(2, 2)	(2, 3)	(2, 4)
(3, 1)	(3, 2)	(3, 3)	(3, 4)
(4, 1)	(4, 2)	(4, 3)	(4, 4)

```
for(int i=1;i<=n;i++)
{
    if (i+y-x>0 && i+y-x<=n)
        cout<<"("<<i<<", "<<i+y-x<<)"<<" ";
}
cout<<endl;
for(int i=n;i>=1;i--)
{
    if (x+y-i>0 && x+y-i<=n)
        cout<<"("<<i<<", "<<x+y-i<<)"<<" ";
}
```

例2：斜线输出

```
3
2 4 9
0 7 4
2 8 5
```

```
2 0 8 2 7 5 4 4 9
```

(1, 1)	(1, 2)	(1, 3)	(1, 4)
(2, 1)	(2, 2)	(2, 3)	(2, 4)
(3, 1)	(3, 2)	(3, 3)	(3, 4)
(4, 1)	(4, 2)	(4, 3)	(4, 4)

要解决的问题：

1. 斜线的走向
2. 斜线开始的位置
3. 斜线结束的位置

问题分析

■ 下半区

初值: $i=k(4,3,2,1), j=1$;

斜线走向: $i++, j++$;

结束: $i>4$;

①	(1, 1)	(1, 2)	(1, 3)	(1, 4)
②	(2, 1)	(2, 2)	(2, 3)	(2, 4)
③	(3, 1)	(3, 2)	(3, 3)	(3, 4)
④	(4, 1)	(4, 2)	(4, 3)	(4, 4)

```
for (int k=n;k>=1;k--)  
{  
    int i=k,j=1;  
    while (i<=n) cout <<a[i++][j++]<<" ";  
}
```

问题分析

■ 上半区

初值: $i=1, j=k(2,3,4)$;

斜线走向: $i++, j++$;

结束: $j>4$;

②				③				④			
(1, 1)	(1, 2)	(1, 3)	(1, 4)	(2, 1)	(2, 2)	(2, 3)	(2, 4)	(3, 1)	(3, 2)	(3, 3)	(3, 4)
(4, 1)	(4, 2)	(4, 3)	(4, 4)								

```
for (int k=2;k<=n;k++)  
{  
    for (int i=1,j=k;j<=n;i++,j++) cout <<a[i][j]<<" ";  
}
```

举一反三

(1, 1)	(1, 2)	(1, 3)	(1, 4)
(2, 1)	(2, 2)	(2, 3)	(2, 4)
(3, 1)	(3, 2)	(3, 3)	(3, 4)
(4, 1)	(4, 2)	(4, 3)	(4, 4)

要解决的问题：

1. 斜线的走向
2. 斜线开始的位置
3. 斜线结束的位置

例3：回型方阵

- 输入 n ，输出 $n*n$ 的回型方阵。
- $n=7$

1	1	1	1	1	1	1
1	2	2	2	2	2	1
1	2	3	3	3	2	1
1	2	3	4	3	2	1
1	2	3	3	3	2	1
1	2	2	2	2	2	1
1	1	1	1	1	1	1

模拟法填充？

方法一

- 红对角线 $i + j = n + 1$
- 蓝对角线 $i = j$
- 两条对角线划分为4个区域:

- ① $i + j \leq n + 1$ 且 $i \leq j$, $a[i][j] = i$
- ② $i + j \leq n + 1$ 且 $i > j$, $a[i][j] = j$
- ③ $i + j > n + 1$ 且 $i > j$, $a[i][j] = n + 1 - i$
- ④ $i + j > n + 1$ 且 $i \leq j$, $a[i][j] = n + 1 - j$

1	1	1	1	1	1	1
1	2	2	2	2	2	1
1	2	3	3	3	2	1
1	2	3	4	3	2	1
1	2	3	3	3	2	1
1	2	2	2	2	2	1
1	1	1	1	1	1	1

用二重循环将二维数组扫描一遍就能完成填充！

方法二

- 用圈号填充
- 求圈号

$a[i][j] = \text{Min}(i, j, n+1-i, n+1-j);$

```
int Min(int a, int b, int c, int d)
{
    int t = min(a, b);
    int s = min(c, d);
    return min(t, s);
}
```

1	1	1	1	1	1	1
1	2	2	2	2	2	1
1	2	3	3	3	2	1
1	2	3	4	3	2	1
1	2	3	3	3	2	1
1	2	2	2	2	2	1
1	1	1	1	1	1	1

例4：蛇形填充

1	2	6	7
3	5	8	13
4	9	12	14
10	11	15	16

要解决的问题：

1. 斜线的走向
2. 斜线开始的位置
3. 斜线结束的位置

问题分析

1	2	6	7
3	5	8	13
4	9	12	14
10	11	15	16

①	(1, 1)	(1, 2)	(1, 3)	(1, 4)
②	(2, 1)	(2, 2)	(2, 3)	(2, 4)
③	(3, 1)	(3, 2)	(3, 3)	(3, 4)
④	(4, 1)	(4, 2)	(4, 3)	(4, 4)
	⑤	⑥	⑦	

问题分析

(1, 1)	(1, 2)	(1, 3)	(1, 4)
(2, 1)	(2, 2)	(2, 3)	(2, 4)
(3, 1)	(3, 2)	(3, 3)	(3, 4)
(4, 1)	(4, 2)	(4, 3)	(4, 4)

■ 上半区: $k(1, 2, 3, 4)$

◆ k 是奇数

初值: $i=k, j=1$;

斜线走向: $i--, j++$;

结束: $i \geq 1$;

◆ k 是偶数

初值: $i=1, j=k$;

斜线走向: $i++, j--$;

结束: $j \geq 1$;

■ 下半区: $k(3, 2, 1)$

◆ k 是奇数

初值: $i=n, j=n-k+1$;

斜线走向: $i--, j++$;

结束: $j \leq n$;

◆ k 是偶数

初值: $i=n-k+1, j=n$;

斜线走向: $i++, j--$;

结束: $i \leq n$;

上半区填充

```
for (int k=1;k<=n;k++)
{
    if (k%2!=0)
    {
        for (int i=k,j=1;j<=k;i--,j++) a[i][j]=t++;
    }
    else
    {
        for (int i=1,j=k;i<=k;i++,j--) a[i][j]=t++;
    }
}
```

例5：神奇的幻方

描述

幻方是一个很神奇的 $N*N$ 矩阵，它的每行、每列与对角线，加起来的数字和都是相同的。

我们可以通过以下方法构建一个幻方。（阶数为奇数）

- 1.第一个数字写在第一行的中间
- 2.下一个数字，都写在上一个数字的右上方：
 - a.如果该数字在第一行，则下一个数字写在最后一行，列数为该数字的右一列
 - b.如果该数字在最后一列，则下一个数字写在第一列，行数为该数字的上一行
 - c.如果该数字在右上角，或者该数字的右上方已有数字，则下一个数字写在该数字的下方

输入

一个数字 N ($N \leq 20$)

输出

按上方法构造的 $2N-1 * 2N-1$ 的幻方

问题分析

1. 第一个数字写在第一行的中间
2. 下一个数字，都写在上一个数字的右上方：
 - a. 如果该数字在第一行，则下一个数字写在最后一行，列数为该数字的右一列
 - b. 如果该数字在最后一列，则下一个数字写在第一列，行数为该数字的上一行
 - c. 如果该数字在右上角，或者该数字的右上方已有数字，则下一个数字写在该数字的下方

17	24	1	8	15
23	5	7	14	16
4	6	13	20	22
10	12	19	21	3
11	18	25	2	9

例：n=3，所以m=5，第一个数字的位置：i=1，j=n；下一个数字位置：i--，j++

1. $i == 1 \ \&\& \ j < m$, 则 $i = m$, $j++$;
2. $i > 1 \ \&\& \ j == m$, 则 $i--$, $j=1$;
3. $i == 1 \ \&\& \ j == m \ || \ a[i-1][j+1] != 0$, 则 $i++$;

循环什么时候结束？

例6：输出杨辉三角

															1															
														1		1														
													1		2		1													
												1		3		3		1												
											1		4		6		4		1											
										1		5		10		10		5		1										
									1		6		15		20		15		6		1									
								1		7		21		35		35		21		7		1								
							1		8		28		56		70		56		28		8		1							
						1		9		36		84		126		126		84		36		9		1						
					1		10		45		120		210		252		210		120		45		10		1					
				1		11		55		165		330		462		462		330		165		55		11		1				
			1		12		66		220		495		792		924		792		495		220		66		12		1			
		1		13		78		286		715		1287		1716		1716		1287		715		286		78		13		1		
	1		14		91		364		1001		2002		3003		3432		3003		2002		1001		364		91		14		1	

问题分析

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

```
for ( int i = 2; i <= n; i++ )  
    for ( int j = 1; j <= ? ; j++ )
```

递推公式: $a[i][j]=a[i-1][j-1]+a[i-1][j]$
