



信奥算法

二维递推

一维
递推
算法

在一维数组内填表格

二维
递推
算法

在二维数组内填表格

杨辉三角形

输入一个正整数n，打印出一个n行的杨辉三角形。n<10

输入样例：
4

输出样例：
1
1 1
1 2 1
1 3 3 1

输入样例：
6

输出样例：
1
1 1
1 2 1
1 3 3 1
1 4 6 4 1
1 5 10 10 5 1

定义二维数组
yh[10][10]

yh[i][j] 代表
杨辉三角形第i行第j列

杨辉三角形

输入一个正整数n，打印出一个n行的杨辉三角形。n<10

	i=0	i=1	i=2	i=3	i=4	i=5	
i=0	0	0	0	0	0	0	缓冲带
i=1	0	1	0	0	0	0	
i=2	0	1	1	0	0	0	每格都依赖 正上方 和左上方
i=3	0	1	2	1	0	0	
i=4	0	1	3	3	1	0	
i=5	0	1	4	6	4	1	

若i==1
且j==1

$$yh[1][1] = 1$$

初始
条件

杨辉三角形

输入一个正整数n，打印出一个n行的杨辉三角形。n<10

	i=0	i=1	i=2	i=3	i=4	i=5	
i=0	0	0	0	0	0	0	缓冲带
i=1	0	1	0	0	0	0	
i=2	0	1	1	0	0	0	
i=3	0	1	2	1	0	0	
i=4	0	1	3	3	1	0	
i=5	0	1	4	6	4	1	

每格都依赖正上方和左上方

若 $i \geq 2$
且 $j \leq i$

$$yh[i][j] = yh[i-1][j-1] + yh[i-1][j]$$

递推
方程

杨辉三角形 - 代码1

```
3  int n,yh[10][10]; ← 全局数组自动清零
```

```
5  cin>>n;
6  for(int i=1;i<=n;i++)
7      for(int j=1;j<=n;j++){
8          if(i==1&&j==1)
9              yh[i][j]=1;
10         else if(i>=2&&j<=i)
11             yh[i][j]=yh[i-1][j-1]+yh[i-1][j];
12     }
```

逐格
填表

```
13 for(int i=1;i<=n;i++){
14     for(int j=1;j<=i;j++)
15         cout<<yh[i][j]<<" ";
16     cout<<endl;
17 }
```

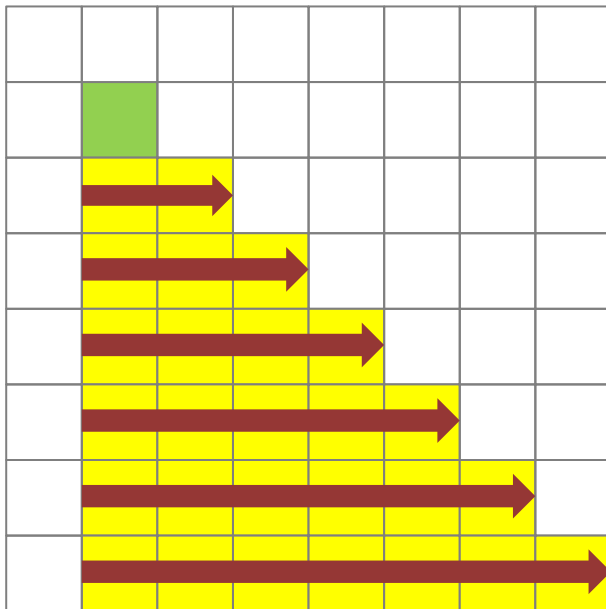
输出
表格

杨辉三角形 - 代码2

```
5  cin>>n;  
6  yh[1][1]=1;  
7  for(int i=2;i<=n;i++)  
8      for(int j=1;j<=i;j++)  
9      yh[i][j]=yh[i-1][j-1]+yh[i-1][j];
```

先枚举行号

再枚举列号



能否先枚举列再枚举行?

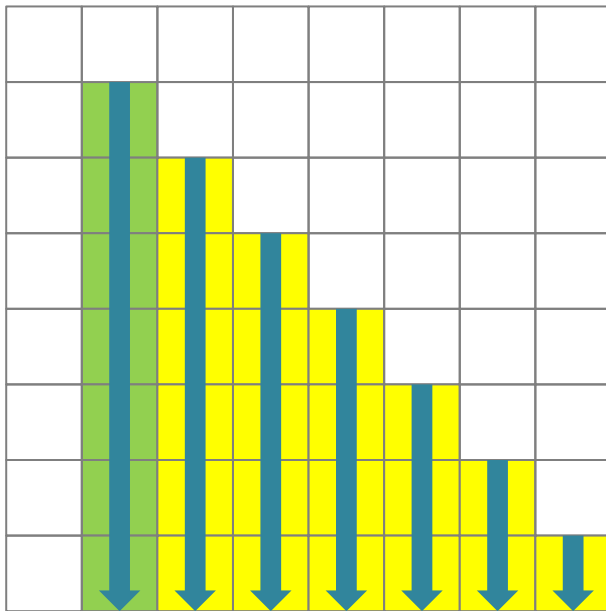
杨辉三角形 - 代码3

6
7
8
9

```
for(int i=1;i<=n;i++)yh[i][1]=1;  
for(int j=2;j<=n;j++)  
    for(int i=j;i<=n;i++)  
        yh[i][j]=yh[i-1][j-1]+yh[i-1][j];
```

先枚举列号

再枚举行号



组合数

从 n 个不同物体中取出 m 个的所有组合
有多少种取法，叫做组合数 $C(n, m)$
输入 n 和 m ，输出 $C(n, m)$

输入样例：

4 2

输入样例：

5 3

输入样例：

6 3

输出样例：

6

输出样例：

10

输出样例：

20

每一格都依赖左上方和正上方邻居

第一列不满足上述依赖关系，特殊处理

$C(i, j)$	$j=0$	$j=1$	$j=2$	$j=3$	$j=4$
$i=0$	1	0	0	0	0
$i=1$	1	1	0	0	0
$i=2$	1	2	1	0	0
$i=3$	1	3	3	1	0
$i=4$	1	4	6	4	1

依赖关系确定后
请设计填表顺序

组合数：递推公式

$$C[0][0] = 1$$

初始
条件

若 $i \geq 1$
且 $j \leq i$

$$C[i][j] = C[i-1][j-1] + C[i-1][j]$$

递推
方程

i个物体
取出j个
共几种

=

i-1个物体
取出j-1个
共几种

+

i-1个物体
取出j个
共几种

分类
讨论

加法
原理

第i个物体
被取出的情况

第i个物体
不被取出的情况

<pre> 1 #include<bits/stdc++.h> 2 using namespace std; 3 int n,m,C[100][100]; 4 int main(){ 5 freopen("combination.in","r",stdin); 6 freopen("combination.out","w",stdout); 7 cin>>n>>m; 8 for(int i=0;i<=n;i++)C[i][0]=1; 9 for(int j=1;j<=n;j++) 10 for(int i=j;i<=n;i++) 11 C[i][j]=C[i-1][j-1]+C[i-1][j]; 12 for(int i=0;i<=n;i++,cout<<endl) 13 for(int j=0;j<=i;j++)cout<<C[i][j]<<" "; 14 cout<<C[n][m]<<endl; 15 return 0; 16 }</pre>	<div style="border: 1px solid black; padding: 5px; margin-bottom: 10px;">0号列初始化</div> <div style="border: 1px solid black; padding: 5px; margin-bottom: 10px;">先枚举列号j 再枚举行号i</div> <div style="border: 1px solid black; padding: 5px;">打印整张表格</div>
---	---

老师用文件输入50 50
同学观察文件输出的表格内容

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
1 15 105 455 1365 3003 5005 6435 6435 5005 3003 1365 455 105 15 1
1 16 120 560 1820 4368 8008 11440 12870 11440 8008 4368 1820 560 120 16 1
1 17 136 680 2380 6188 12376 19448 24310 24310 19448 12376 6188 2380 680 136 17 1
1 18 153 816 3060 8568 18564 31824 43758 48620 43758 31824 18564 8568 3060 816 153 18 1
1 19 171 969 3876 11628 27132 50388 75582 92378 92378 75582 50388 27132 11628 3876 969 171 19 1
1 20 190 1140 4845 15504 38760 77520 125970 167960 184756 167960 125970 77520 38760 15504 4845 1140 190 20 1
1 21 210 1330 5985 20349 54264 116280 293930 352716 352716 293930 116280 54264 20349 5985 1330 210 21 1
1 22 231 1540 7315 26334 74613 170544 319770 497420 646646 705432 646646 497420 319770 170544 74613 26334 7315 1540 231 22 1
1 23 253 1771 8855 33649 100947 245157 490314 817190 1144066 1352078 1352078 1144066 817190 490314 245157 100947 33649 8855 1771 253 23 1
1 24 276 2024 10626 42504 134596 346104 735471 1307504 1961256 2496144 2704156 2496144 1961256 1307504 735471 346104 134596 42504 10626 2024 276 24 1
1 25 300 2300 12650 53130 177100 480700 1081575 2042975 3268760 4457400 5200300 5200300 4457400 3268760 2042975 1081575 480700 177100 53130 12650 2300 300 25 1
1 26 325 2600 14950 65780 230230 657800 1562275 3124550 5311735 7726160 9657700 10400600 9657700 7726160 5311735 3124550 1562275 657800 230230 65780 14950 2600 325
1 27 351 2925 17550 80730 296010 888030 2220075 4686825 8436285 13037895 17383860 20058300 20058300 17383860 13037895 8436285 4686825 2220075 888030 296010 80730 17550 351
1 28 378 3276 20475 98280 376740 1184040 3108105 6906900 13123110 21474180 30421755 37442160 40116600 37442160 30421755 21474180 13123110 6906900 3108105 1184040 376740 98280 378
1 29 406 3654 23751 118755 475020 1560780 4292145 10015005 20030010 34597290 51895935 67863915 77558760 77558760 67863915 51895935 34597290 20030010 10015005 4292145 1560780 475020 406
1 30 435 4060 27405 142506 593775 2035800 5852925 14307150 30045015 54627300 86493225 119759850 145422675 155117520 145422675 119759850 86493225 54627300 30045015 5852925 2035800 593775 435
1 31 465 4495 31465 169911 736281 2629575 7888725 20160075 44352165 84672315 141120525 206253075 265182525 300540195 300540195 265182525 206253075 141120525 84672315 20160075 7888725 2629575 465
1 32 496 4960 35960 201376 906192 3365856 10518300 28048800 64512240 129024480 225792840 347373600 471435600 565722720 601080390 565722720 471435600 347373600 225792840 129024480 64512240 28048800 10518300 3365856 496
1 33 528 5456 40920 237336 1107568 4272048 13884156 38567100 92561040 193536720 354817320 573166440 818809200 1037158320 1166803110 1166803110 1037158320 818809200 573166440 354817320 193536720 92561040 38567100 13884156 528
1 34 561 5984 46376 278256 1344904 5379616 18156204 52451256 131128140 286097760 548354040 927983760 1391975640 1855967520 -2091005866 -1961361076 -2091005866 1855967520 927983760 548354040 286097760 131128140 52451256 18156204 5379616 1344904 561
1 35 595 6545 52360 324632 1623160 6724520 23535820 70607460 183579396 417225900 834451800 1476337800 -1975007896 -1047024136 -235038346 242600354 242600354 -235038346 -1047024136 -1975007896 1476337800 834451800 417225900 183579396 70607460 23535820 6724520 1623160 324632 52360 595

建议使用
long long储存

若long long
也储存不下结果

溢出错误

常见出题方式
对结果取模后
方便测试打分

快快编程
kkcoding.net

组合数

从 n 个不同物体中取出 m 个的所有组合
有多少种取法，叫做组合数 $C(n, m)$

输入 n 和 m ， $0 \leq m \leq n \leq 1000$

输出 $C(n, m)$ 对1000000007取模

输入样例：

4 2

输入样例：

5 3

输入样例：

6 3

输出样例：

6

输出样例：

10

输出样例：

20

```

1  #include<bits/stdc++.h>
2  using namespace std;
3  typedef long long ll;
4  const ll MOD=1000000007;
5  const ll N=1009;
6  ll n,m,C[N][N];
7  int main(){
8      cin>>n>>m;
9      for(ll i=0;i<=n;i++)C[i][0]=1;
10     for(ll j=1;j<=n;j++)
11         for(ll i=j;i<=n;i++)
12             C[i][j]=(C[i-1][j-1]+C[i-1][j])%MOD;
13     cout<<C[n][m]<<endl;
14     return 0;
15 }

```

简写ll

模数定义为常数

每步计算都取模
防止中间结果溢出

记笔记

高频错误

忘记中间步骤取模

棋盘路径计数 - 无障碍

在棋盘格上，小明站在第1行第1列的位置(左上角)，罗马在第n行第m列的位置(右下角)。小明每一步只可以向右走一格或者向下走一格。输入n和m ($1 \leq m, n \leq 20$)，输出小明有多少种方法可以走到罗马。

输入样例：

2 2

输入样例：

4 3

输出样例：

2

输出样例：

10

棋盘路径计数 - 无障碍

$f[i][j]$ 表示 到第*i*行第*j*列有多少种走法

	j=0	j=1	j=2	j=3	j=4	
i=0	0	0	0	0	0	← 缓冲带
i=1	0	1	1	1	1	
i=2	0	1	2	3	4	
i=3	0	1	3	6	10	
i=4	0	1	4	10	20	

↑
缓冲带

$$f[1][1] = 1$$

(i,j)不是(1,1)时 $f[i][j] = f[i-1][j] + f[i][j-1]$

初始
条件

递推
方程

棋盘路径计数 - 无障碍

```
1  #include<iostream>
2  #define M 29
3  using namespace std;
4  long long f[M][M],n,m;
5  int main(){
6      cin>>n>>m;
7      for(int i=1;i<=n;i++)
8          for(int j=1;j<=m;j++)
9              if(i==1&&j==1)
10                 else f[i][j]=
11  cout<<<<endl;
12  return 0;
13 }
```

棋盘路径计数 - 有障碍

在棋盘格上，小明站在第1行第1列的位置（左上角），目的地在第5行第5列的位置（右下角）。小明每一步只可以向右走一格或者向下走一格。

输入5行5列的棋盘，0表示无障碍可以通行，#表示此处有障碍，输出小明有多少种方法可以走到目的地。

输入样例：

00000

##0##

##0##

##000

##000

输出样例：

3

输入样例：

00000

0##0#

0##00

0###0

00000

输出样例：

2

```
char d[6][6];
```

数组d保存整张地图

$d[i][j]$ 表示第*i*行第*j*列的字符

$f[i][j]$ 表示从左上角走到第*i*行第*j*列有多少种走法

```
1 /*
2 f[i][j]表示从左上角走到
3 第i行第j列有多少种走法
4 输入地图:
```

```
5 00000
6 0000#
7 #0#0#
8 00#00
9 #0000
```

```
10 f[i][j]手算表格
```

```
11      j=0,j=1,j=2,j=3,j=4,j=5
```

```
12 i=0      0      0      0      0      0      0
```

```
13 i=1      0      1      1      1      1      1
```

```
14 i=2      0      1      2      3      4      0
```

```
15 i=3      0      

|  |
|--|
|  |
|  |
|  |


```

```
16 i=4      0
```

```
17 i=5      0
```

```
18 */
```

← 请写出这5个数

$f[i][j]$ 表示从左上角到第*i*行第*j*列有几种走法

输入:

```
00000
0000#
#0#0#
00#00
#0000
```

	j=0	j=1	j=2	j=3	j=4	j=5
i=0	0	0	0	0	0	0
i=1	0	1	1	1	1	1
i=2	0	1	2	3	4	0
i=3	0	0	2	0	4	0
i=4	0	0	2	0	4	4
i=5	0	0	2	2	6	10

初始
条件

$$f[1][1] = (d[1][1] == '0')$$

递推
方程

(i,j)
不是
(1,1)

若 $d[i][j]$ 是'#'

$$f[i][j] = 0$$

若 $d[i][j]$ 是'0'

$$f[i][j] = f[i-1][j] + f[i][j-1]$$

$f[i][j]$ 表示从左上角到第*i*行第*j*列有几种走法

```
23 char d[6][6];
24 for(int i=1;i<=5;i++)
25     for(int j=1;j<=5;j++)
26         cin>>d[i][j];
27 for(int i=1;i<=5;i++)
28     for(int j=1;j<=5;j++){
29         if(i==1&&j==1) f[1][1]=;
30         else if() f[i][j]=0;
31         else f[i][j]=;
32     }
33 cout<<f[5][5]<<endl;
```

```
10 f[i][j] 手算表格
11     j=0, j=1, j=2, j=3, j=4, j=5
12 i=0    0    0    0    0    0    0
13 i=1    0    1    1    1    1    1
14 i=2    0    1    2    3    4    0
15 i=3    0    0    2    0    4    0
16 i=4    0    0    2    0    4    4
17 i=5    0    0    2    2    6    10
18 */
```

打印
表格



对照
手算
结果



定位
错误

```
33 cout<<f[5][5]<<endl;
34 for(int i=0;i<=5;i++,cout<<endl)
35     for(int j=1;j<=5;j++)cout<<f[i][j]<<" ";
```


现场挑战

快快编程311

快快编程
kkcoding.net

```
1 /*
2 g[i][j]表示从左上角走到
3 第i行第j列最多有多少金币
4 输入地图:
```

```
5 *****
```

```
6 *0000
```

```
7 *1111
```

```
8 *2222
```

```
9 *****
```

```
10 g[i][j]手算表格
```

```
11      j=0, j=1, j=2, j=3, j=4, j=5
```

```
12 i=0 -INF -INF -INF -INF -INF -INF
```

```
13 i=1 -INF -1 -2 -3 -4 -5
```

```
14 i=2 -INF -2 -2 -2 -2 -2
```

```
15 i=3 -INF -3 -1 0 1 2
```

```
16 i=4 -INF -4 1 3
```

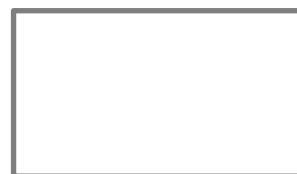
```
17 i=5 -INF -5 0 2
```

```
18 */
```

请在程序开头写出
表格含义+手算表格

完成后给老师检查

请写出
这4个数



快快编程
kkcoding.net

x[i][j]表示第**i**行第**j**列加減的金币数

输入:

*0000

*1111

*2222

	j=0	j=1	j=2	j=3	j=4	j=5
i=0						
i=1		-1	-1	-1	-1	-1
i=2		-1	0	0	0	0
i=3		-1	1	1	1	1
i=4		-1	2	2	2	2
i=5		-1	-1	-1	-1	-1

28

29

30

31

32

33

34

```
for(int i=1;i<=5;i++)
    for(int j=1;j<=5;j++){
        char ch;
        cin>>ch;
        if(ch=='*') x[i][j]=-1;
        else x[i][j]=ch-'0';
    }
```

$g[i][j]$ 表示从左上角走到
第*i*行第*j*列最多有多少金币

输入:

*0000

*1111

*2222

	j=0	j=1	j=2	j=3	j=4	j=5
i=0	-INF	-INF	-INF	-INF	-INF	-INF
i=1	-INF	-1	-2	-3	-4	-5
i=2	-INF	-2	-2	-2	-2	-2
i=3	-INF	-3	-1	0	1	2
i=4	-INF	-4	1	3	5	7
i=5	-INF	-5	0	2	4	6

初始
条件

$$f[1][1] = x[1][1]$$

递推
方程

(i,j)
不是
(1,1)

$$g[i][j] = \max(g[i-1][j], g[i][j-1]) + d[i][j]$$

为什么缓冲带
要填-INF

让从缓冲带进入地图的非法
路径不可能破坏最优解

$g[i][j]$ 表示从左上角走到
第*i*行第*j*列最多有多少金币

$x[i][j]$ 表示第*i*行第*j*列加减的金币数

```
21 typedef long long ll;
22 const ll INF=100;
23 const ll N=10;
24 ll g[N][N],x[N][N];

35 for(ll i=0;i<=5;++i) g[i][0]=-INF;
36 for(ll j=0;j<=5;++j) g[0][j]=
37 for(ll i=1;i<=5;i++)
38     for(ll j=1;j<=5;j++){
39         if(i==1&& j==1)g[1][1]=x[1][1];
40         else g[i][j]=
41     }
42 cout<<g[5][5]<<endl;
```

缓冲带的使用

优点：简化递推程序
将递推初始化和递推方程统一处理

易错点：缓冲带的数值初始化错误
提供了“原本不存在”的行走路径
影响了正常递推的运算

例如：快快311的缓冲带不该填0

作业要求

写程序前请写明：

1. 二维数组每一格的含义
2. 手算样例对应表格

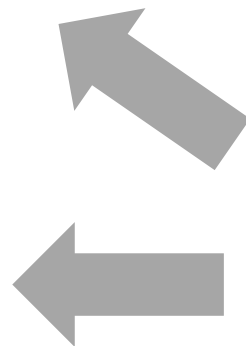
查错方法：

1. 打印二维数组
2. 和手算表格对比找不同

```
1  /*
2  f[i][j]代表
3      j=0,1,2,3,4,5
4  i=0
5  i=1
6  i=2
7  i=3
8  i=4
9  i=5
10 */
```



模版格式
供参考



快快编程
kkcoding.net

调试查错



快快编程作业

1779

293

311

拓展题

312, 313