# 实验 循环和数组

樊子谦

软件学院 软件工程 2024级1班

**Exercise 1：Please write programs to output following special figures.**

**Task 1:**

Input an integer *n*, output *n* characters ‘\*’ in one line.

#include <bits/stdc++.h>

using namespace std;

int main() {

int n;

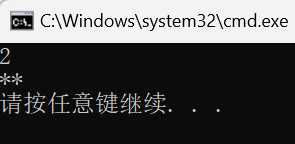
cin >> n;

while (n--)

cout << '\*';

return 0;

}



**Task 2:**

Input an integer *n*, output *n\*n* matrix made of characters ‘\*’.

#include <bits/stdc++.h>

using namespace std;

int main() {

int n;

cin >> n;

for (int i = 1; i <= n; i++) {

for (int j = 1; j <= n; j++) {

cout << '\*';

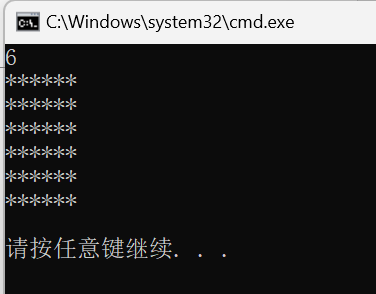
}

cout << '\n';

}

return 0;

}



**Task 3:**

Input an integer *n*, output a triangle made of characters ‘\*’.

#include <bits/stdc++.h>

using namespace std;

int main() {

int n;

cin >> n;

for (int i = 1; i <= n; i++) {

for (int j = 1; j <= i; j++) {

cout << '\*';

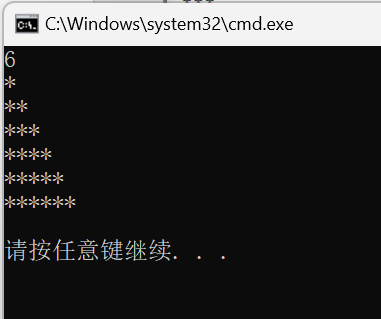
}

cout << '\n';

}

return 0;

}



**Task 4:**

Input an integer *n*, output a triangle made of characters ‘\*’.

#include <bits/stdc++.h>

using namespace std;

int main() {

int n;

cin >> n;

for (int i = 1; i <= n; i++) {

/\*

for(int j = 1; j <= n - i; j++) cout << ' ';

for(int j = 1; j <= i \* 2 - 1; j++) cout << '\*';

\*/

string empty(n - i, ' ');

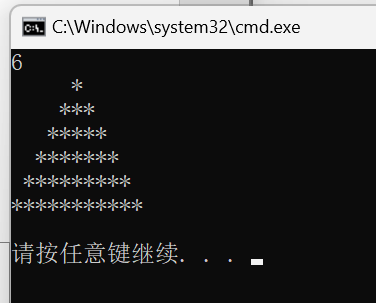
string stars(i \* 2 - 1, '\*');

cout << empty << stars << '\n';

}

return 0;

}



普通的双重循环实现在注释里面。

这里利用了一些小trick，即先观察每一行开头的空格出现的规律：第一行是n-1个，第二行是n-2个……而星号也一样，以1、3、5、7这样的等差数列递增，所以直接利用字符串，把每一行出现的这些字符初始化为一个字符串然后输出，就不用多嵌套一层循环啦！这个方法后面也会用到。

***字符串string的初始化规则：第一个参数为长度，第二个参数为填充的字符。***

**Exercise 2：Please write programs to output following special figures.**

**Task 1:**

Input an integer *n*, output 1~*n* in one line.

#include <bits/stdc++.h>

using namespace std;

int main() {

int n;

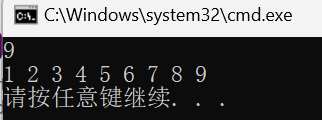
cin >> n;

for (int i = 1; i <= n; i++)

cout << i << ' ';

return 0;

}



**Task 2:**

Input an integer *n*, output *n\*n* matrix made of numbers 1~*n*.

#include <bits/stdc++.h>

using namespace std;

int main() {

int n;

cin >> n;

for (int i = 1; i <= n; i++) {

for (int j = 1; j <= n; j++) {

cout << j << ' ';

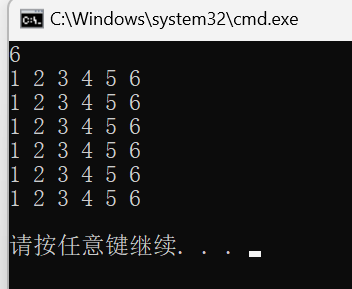
}

cout<<'\n';

}

return 0;

}



**Task 3:**

Input an integer *n*, output *n\*n* matrix made of numbers 1~*n\*n*.

最简单的思路就是引入一个计数器，每次输出一个就+1.

#include <bits/stdc++.h>

using namespace std;

int main() {

int n,cnt=0;

cin >> n;

for (int i = 1; i <= n; i++) {

for (int j = 1; j <= n; j++) {

cnt++;

cout << cnt << ' ';

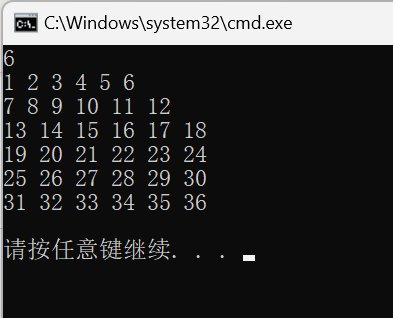
}

cout<<'\n';

}

return 0;

}



**Task 4:**

Input an integer *n*, output a triangle made of numbers.

#include <bits/stdc++.h>

using namespace std;

int main() {

int n,cnt=0;

cin >> n;

for (int i = 1; i <= n; i++) {

string empty(n - i, ' ');

cout << empty;

for (int j = 1; j <= i; j++) {

cnt++;

cout << cnt << ' ';

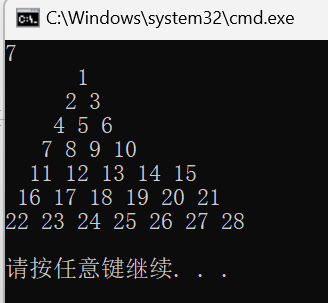
}

cout<<'\n';

}

return 0;

}



和上面的星星三角形的实现一样。但是因为十位数的宽度和个位数的宽度不同，这个三角形很丑。似乎可以用setw控制输出更整齐，但是这样输出就和样例不符了。

**Task 5:**

Input an integer *n*, output *n\*n* matrix made of numbers 1~*n*.

For example, when *n* is 3, The output is

1 2 3

2 3 1

3 1 2

When *n* is 4, The Output is

1 2 3 4

2 3 4 1

3 4 1 2

4 1 2 3

每一行开始的数字轮换。

开始的时候是这样的想法：生成一个类似“1 2 3 4 1 2 3 4”这样翻了一倍的列表，然后i遍历1234，j一样，输出列表中的第i+j-1个元素。

但这样就不好玩了，因为开了2n的空间。作为一名数学嗅觉比较敏锐的学生（不），我们捕捉到了这样的特征：

1. 每一个元素的大小都和i、j都有关； （2）元素的形式是轮换形式

所以很容易得到每一个元素符合(i+j-2)%n+1这个式子，写程序就很简单了。

#include <bits/stdc++.h>

using namespace std;

int main() {

int n;

cin >> n;

for (int i = 1; i <= n; i++) {

for (int j = 1; j <= n; j++) {

cout<<(i+j-2)%n+1<<' ';

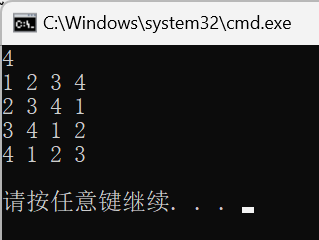
}

cout<<'\n';

}

return 0;

}



**Exercise 3: Debugging.**

题目给了一个非常丑陋的程序来改错，而且丑陋得有点惨不忍睹。

好在给了样例输出，我们大概理解了题目的要求：

实现一个程序，里面有双重循环嵌套。i从1遍历到2，j从0遍历到3.

每一个i和j都要输出：i+j i-j i\*j i^j （后面两项j＝0的时候不用）i/j i%j

（吐槽：这个^居然不是按位异或……看了大半天没整明白）

原本的代码是这样的，我们加上行数方便指出错误：

1. #include <iostream>

2. #include <iomanip>

3. using namespace std;

4.

5. int main()

6. {

7.         int i = 1;

8.         double a;

9.         double b;

10.

11.         cout << setprecision( 2 );

12.

13.         for ( int i; i <= 2; i++ )

14.                 cout << "i is now equal to " << i << endl;

15.

16.         for ( int j; j <= 3; j++ )

17.         {

18.                 cout << "\tj is now equal to " << j << endl;

19.

20.                 cout << "\t\ti + j = " << i + j << "\ti - j = "

21.                          << i - j << endl;

22.                 cout << "\t\ti \* j = " << i \* j << "\ti ^ j = "

23.                          << pow( i, j ) << endl;

24.

25.                 if ( j = 0 )

26.                         continue;

27.                 else

28.                 {

29.                         a = i;

30.                         b = j;

31.                         cout << "\t\ti / j = " << a / b

32.                                         "\ti % j = " << a % b << endl;

33.                 } // end else

34.         } // end for

35.

36.         cout << "\nThe final values of i and j are: " << i

37.         << " and " << j << endl;

38. } // end main

查错：

37行提示j不存在。发现这个代码本来想在最后输出i和j，但是只在循环体内部定义了i和j。这样定义一定是错误的。因为在for的作用域里面，i和j是独立的，并不影响作用域外，包括末尾的输出。即使在第七行i有定义，也没法得到期望的输出。

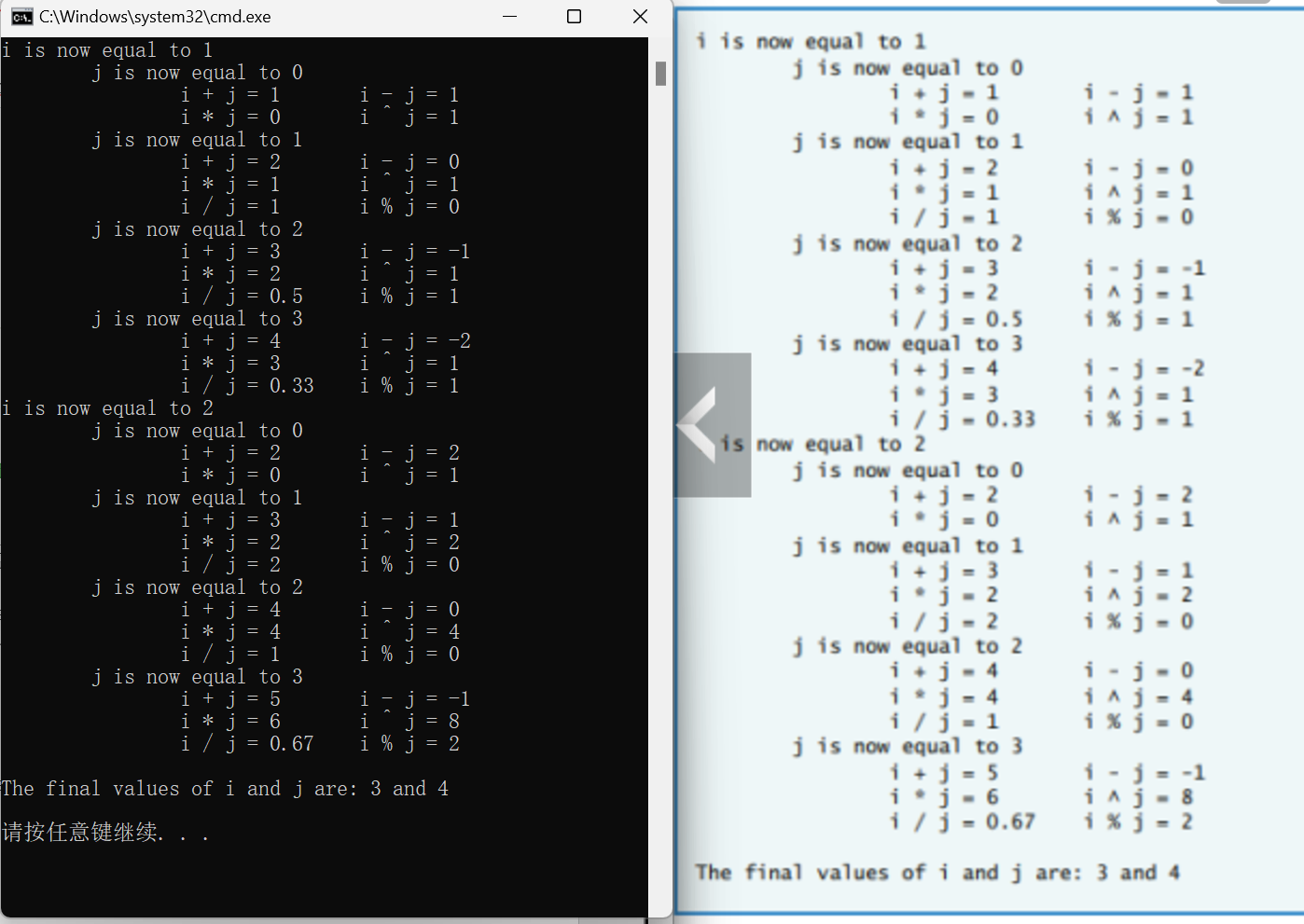
把第七行的j加上，第13行、16行的for()里面的int去掉，不能重新定义一遍。13行i要赋初值1，16行j赋初值0；

两层循环缺了个大括号，变成独立的两个循环了，加上。注意输出final result的一行要在大括号外面，之前扩进去了还没发现。

第32行，浮点数之间是不能取模的，要用fmod或者转类型为int再运算。这里用fmod。

22行调用pow函数却没有引用cmath头文件，补上。

修改完成以后，输出如下，与标准答案一致。



正确代码如下：

#include <cmath>

#include <iomanip>

#include <iostream>

using namespace std;

int main() {

int i, j;

double a;

double b;

cout << setprecision(2);

for (i = 1; i <= 2; i++) { // 初始化i

cout << "i is now equal to " << i << endl;

for (j = 0; j <= 3; j++) {

cout << "\tj is now equal to " << j << endl;

cout << "\t\ti + j = " << i + j << "\ti - j = " << i - j << endl;

cout << "\t\ti \* j = " << i \* j << "\ti ^ j = " << pow(i, j) << endl;

if (j == 0)

continue;

else {

a = i;

b = j;

cout << "\t\ti / j = " << a / b << "\ti % j = " << fmod(a, b) << endl;

}

}

}

cout << "\nThe final values of i and j are: " << i << " and " << j << endl;

return 0;

}

心得：

这次的任务虽然也很简单，但是可以发现，对于一些任务，多加思考会想到一些实现起来更加简单或者复杂度更优秀的算法。在动手敲代码之前一定要多动脑思考，写起来才会更简单。