

循环结构

for 循环通常用在有固定循环次数的循环语句中。

while 语句用在满足某种条件时才循环的语句中，先判断条件是否满足再执行语句，所以。

do-while 语句和 while 基本相同，所不同的是，它先执行循环语句，再判断条件是否满足，也就是说，循环语句至少能执行一次。

常用词语：

num 数字

sum 总数、金额

score 次数、分数

ans answer 回答、答案

min 最小数

max 最大数

gcd 最大公约数

lcm 最小公倍数

tmp temp 临时（用完就丢）

for 语句

3.1-1

输出数字 1、2、3、4、5、6。

```
#include<iostream>
using namespace std;
int main()
{
    cout<<1;
    cout<<2;
    cout<<3;
    cout<<4;
    cout<<5;
    return 0;
}
```

1 至 6 可以这样写, 1 至 1000 怎么写?
写一万行代码?

3.1-2

从 1 开始

```
#include<iostream>
using namespace std;
int main()
{
    int i;
    for(i=1;i<=1000;i=i+1)
    {
        cout<<i<<" ";
    }
    return 0;
}
```

到 1000 结束

i=i+1 每次递增 1

i 在这里不是 1 个数字, 而是很多数字。
在本段代码中, 是 1、2、3、4、5……一直到 10000 这些数字。

3.1-3

```
#include<iostream>
using namespace std;
int main()
{
    int i;
    for(i=3;i<=12;i=i+1)
    {
        cout<<i<<" ";
    }
    return 0;
}
```

从 3 开始 到 12 结束 每次递增 1

修改数字，编译运行，观察输出数字的变化，体会代码的作用。

这句代码主要功能是限制程序运行几次，3-12 表示运行 10 次；

如果改成 `for(i=1;i<=6;i=i+1)`，是运行 6 次；

如果改成 `for(i=1;i<=6;i=i+2)`，是运行 3 次；

3.1-4

```
#include<iostream>
using namespace std;
int main()
{
    int i;
    for(i=3;i<=12;i=i+2)
        cout<<i<<" ";
    return 0;
}
```

本段代码只修改 1 个数字，编译运行，观察输出数字的变化，体会代码的作用。

`i=i+2` 每次递增 2.

把 2 修改成 3、4 等数字观察一下，体会代码作用。

3.1-5

```
#include<iostream>
using namespace std;
int main()
{
    int i;
    for(i=3;i<=12;i++)
        cout<<i<<" ";
    return 0;
}
```

`i=i+1` 通常这样写：`i++`

`<<" "` 作用是产生空格，隔开数字。

3.1-6

```
#include<iostream>
using namespace std;
int main()
{
    int i,a,b;
    cin>>a>>b;
    for(i=a;i<=b;i++)
        cout<<i<<" ";
    return 0;
}
```

输入: 6 15

输出: 6、7、8、9、10、11、12、13、14、
15

3.1-7

```
#include<iostream>
using namespace std;
int main()
{
    int i;
    for (i=1; i<=20; i++)
    {
        if (i%2==0)
            cout<<i<<" ";
    }
    return 0;
}
```

输出 1 至 20 之间的偶数

3.1-8

```
#include<iostream>
using namespace std;
int main()
{
    int i;
    for(i=2;i<=20;i=i+2)
        cout<<i<<" ";
    return 0;
}
```

输出 1 至 20 之间的偶数

3.1-9

输出 1 至 20 之间的奇数

```
#include<iostream>
using namespace std;
int main()
{
    int i;
    for (i=1;i<=20;i++)
    {
        if (i%2!=0)
        {
            cout<<i<<" ";
        }
    }
    return 0;
}
```

3.1-10

输出 1 至 20 之间的奇数

```
#include<iostream>
using namespace std;
int main()
{
    int i;
    for(i=1;i<=20;i=i+2)
        cout<<i<<" ";
    return 0;
}
```

3.1-11

输出 1 至 20 之间 3 的倍数

```
#include<iostream>
using namespace std;
int main()
{
    int i;
    for (i=1; i<=20; i++)
    {
        if (i%3==0)
        {
            cout<<i<<" ";
        }
    }
    return 0;
}
```

3.1-12

输出 1 至 20 之间 3 的倍数

```
#include<iostream>
using namespace std;
int main()
{
    int i;
    for(i=3; i<=20; i=i+3)
        cout<<i<<" ";
    return 0;
}
```

3.1-13

输出 1 至 50 之间 3、5 的公倍数

```
#include<iostream>
using namespace std;
int main()
{
    int i;
    for (i=1;i<=50;i++)
    {
        if (i%3==0&& i%5==0)
        {
            cout<<i<<" ";
        }
    }
    return 0;
}
```

15 30 45

3.1-14

输出 30、50 的公约数

```
#include<iostream>
using namespace std;
int main()
{
    int i;

    for (i=1;i<=30;i++)
    {
        if (30%i==0&&50%i==0)
        {
            cout<<i<<" ";
        }
    }
    return 0;
}
```

1 2 5 10

3.1-15

求 1~7 这几个数的和

```
#include<iostream>
```

```
using namespace std;
```

```
int main()
```

```
{
```

```
    int i, sum;
```

```
    sum=0;
```

```
    for(i=1;i<=7;i++)
```

```
    {
```

```
        sum=sum+i;
```

```
    }
```

```
    cout<<sum;
```

```
    return 0;
```

```
}
```

sum 总数

把 0 修改成别的数，观察输出结果变化，体会它的作用。

通常情况下这样写：score+=i

3.1-16

求 1~7 中偶数之和

```
#include<iostream>
```

```
using namespace std;
```

```
int main()
```

```
{
```

```
    int i, sum;
```

```
    sum=0;
```

```
    for(i=1;i<=7;i++)
```

```
    {
```

```
        if (i%2==0)
```

```
        {
```

```
            sum+=i; ————— sum=sum+i
```

```
        }
```

```
    }
```

```
    cout<<sum;
```

```
    return 0;
```

```
}
```

3.1-17

```
#include<iostream>
using namespace std;
int main()
{
    int i,ou,ji;
    ou=0;// 偶数的和
    ji=0;// 奇数的和
    for(i=1;i<=7;i++)
    {
        if (i%2==0)
            ou+=i;// 本句等于 ou=ou+i
        else
            ji+=i;// 本句等于 ji=ji+i
    }
    cout<<ou<<" "<<ji;
    return 0;
}
```

求 1~7 中偶数、奇数之和。

3.1-18

```
#include<iostream>
using namespace std;
int main()
{
    int i,ou,ji;
    ou=0;// 偶数个数
    ji=0;// 奇数个数
    for (i=1;i<=7;i++)
    {
        if (i%2==0) ou++; //ou++ 就是 ou=ou+1
        else        ji++; //ji++ 就是 ji=ji+1
    }
    cout<<ou<<" 个偶数 "<<" "<<ji<<" 个奇数 ";
    return 0;
}
```

求 1~7 中偶数、奇数个数。

3.1-19

```
#include<iostream>
using namespace std;
int main()
{
    int i, ou, ji, s1, s2;
    ou=0; // 偶数个数
    ji=0; // 奇数个数
    s1=0; // 偶数的和
    s2=0; // 奇数的和
    for (i=1; i<=7; i++) // i 就是输出的数字
    {
        if (i%2==0)
        {
            s1=s1+i; // 偶数的和
            ou=ou+1; // 偶数的个数
            cout<<i<<" "; // 输出偶数
        }
        else
        {
            s2=s2+i; // 奇数的和
            ji=ji+1; // 奇数的个数
        }
    }
    cout<<endl<<ou<<" 个偶数 "<<" "<<ji<<" 个奇数 "<<endl;
    cout<<" 偶数和: "<<s1<<" "<<" 奇数和: "<<s2;
    return 0;
}
```

求 1~7 中偶数、奇数个数，
偶数和，奇数和，并且列出偶数。

```
2 4 6
3 个偶数 4 个奇数
偶数和: 12 奇数和: 16
```

while 语句

3.2-1

```
#include<iostream>
using namespace std;
int main()
{
    int i;
    i=0;
    while(i<10)
    {
        cout<<i<<" ";
        i++;
    }
    cout<<endl<<" 观察: "<<i;
    return 0;
}
```

0 1 2 3 4 5 6 7 8 9

观察: 10

通过本例题，我们观察到由于 $i < 10$ 的限制，while 内循环是 0-9，当运行到 10 时，终止运行。

3.2-2

```
#include<iostream>
using namespace std;
int main()
{
    int i;
    i=0;
    while(i<0)
    {
        cout<<i<<" ";
        i++;
    }
    cout<<endl<<" 观察: "<<i;
    return 0;
}
```

观察: 0

观察本例题，我们知道不符合条件，while 不会运行。

do-while 语句

3.2-3

```
#include<iostream>
using namespace std;
int main()
{
    int i;
    i=0;
    do
    {
        cout<<i<<" ";
        i++;
    }
    while (i<10);
    cout<<endl<<" 观察: "<<i;
    return 0;
}
```

```
0 1 2 3 4 5 6 7 8 9
观察: 10
```

结果与 while 一样，当超出运行条件限制时终止运行。

这两种语句哪里不同？

3.2-4

```
#include<iostream>
using namespace std;
int main()
{
    int i;
    i=0;
    do
    {
        cout<<i<<" ";
        i++;
    }
    while (i<0);
    cout<<endl<<" 观察: "<<i;
    return 0;
}
```

```
0
观察: 1
```

观察本例题，我们知道 do-while 是先执行，当遇到条件限制时终止运行。

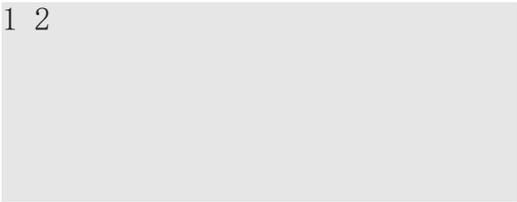
与 3.1-2 相比较观察。

break 和 continue

3.3-1

```
#include<iostream>
using namespace std;
int main()
{
    int i;
    for(i=1;i<=20;i++)
    {
        if (i%3==0)
            break;
        cout<<i<<" ";
    }
    return 0;
}
```

break 和 continue 用于终止循环或者跳过特定语句。

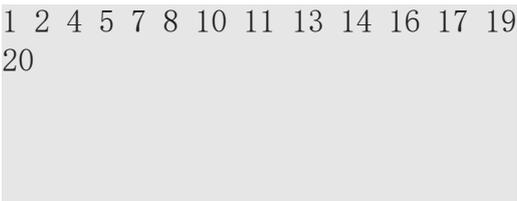


1 2

break, 直接终止 程序。

3.3-2

```
#include<iostream>
using namespace std;
int main()
{
    int i;
    for (i=1;i<=20;i++)
    {
        if (i%3==0)
            continue;
        cout<<i<<" ";
    }
    return 0;
}
```



1 2 4 5 7 8 10 11 13 14 16 17 19
20

continue, 只终止 特定语句。

3.3-3

```
#include<iostream>
using namespace std;
int main()
{
    int i, j;
    for (i=1; i<=9; i++)
    {
        for (j=1; j<=9; j++)
        {
            if (j%3==0) break;
            cout<<j<<" ";
        }
        cout<<endl;
    }
    return 0;
}
```

```
1 2
1 2
1 2
1 2
1 2
1 2
1 2
1 2
1 2
1 2
```

在多层循环中，break 只终止自身所在循环。

3.3-4

```
#include<iostream>
using namespace std;
int main()
{
    int i, j;
    for (i=1; i<=9; i++)
    {
        for (j=1; j<=9; j++)
        {
            if (j%3==0) continue;
            cout<<j<<" ";
        }
        cout<<endl;
    }
    return 0;
}
```

```
1 2 4 5 7 8
1 2 4 5 7 8
1 2 4 5 7 8
1 2 4 5 7 8
1 2 4 5 7 8
1 2 4 5 7 8
1 2 4 5 7 8
1 2 4 5 7 8
1 2 4 5 7 8
1 2 4 5 7 8
```

在多层循环中，continue 只终止特定语句，继续执行下一语句。

多重循环

3.4-1

```
#include<iostream>
using namespace std;
int main()
{
    int i, j;
    for (i=1; i<=6; i++)
    {
        for (j=1; j<=6; j++)
        {
            cout<<"*";
        }
        cout<<endl; // 换行
    }
    return 0;
}
```

```
*****
*****
*****
*****
*****
*****
```

改成: cout<<8<<" ";

```
888888
888888
888888
888888
888888
888888
```

3.4-2

```
#include<iostream>
using namespace std;
int main()
{
    int i, j;
    for (i=1; i<=6; i++)
    {
        for (j=1; j<=6; j++)
        {
            cout<<j<<" ";
        }
        cout<<endl;
    }
    return 0;
}
```

```
1 2 3 4 5 6
1 2 3 4 5 6
1 2 3 4 5 6
1 2 3 4 5 6
1 2 3 4 5 6
1 2 3 4 5 6
```

j 改成 i

```
1 1 1 1 1 1
2 2 2 2 2 2
3 3 3 3 3 3
4 4 4 4 4 4
5 5 5 5 5 5
6 6 6 6 6 6
```

3.4-3

```
#include<iostream>
```

```
using namespace std;
```

```
int main()
```

```
{
```

```
    int i, j;
```

```
    for (i=1; i<=6; i++)
```

* 改成 j

```
    {
```

```
        for (j=1; j<=i; j++)
```

```
        {
```

```
            cout<<"*"<<" ";
```

* 改成 i

```
        }
```

```
        cout<<endl;
```

```
    }
```

```
    return 0;
```

```
}
```

```
*
* *
* * *
* * * *
* * * * *
* * * * * *
```

```
1
1 2
1 2 3
1 2 3 4
1 2 3 4 5
1 2 3 4 5 6
```

```
1
2 2
3 3 3
4 4 4 4
5 5 5 5 5
6 6 6 6 6 6
```

3.4-4

```
#include<iostream>
```

```
using namespace std;
```

```
int main()
```

```
{
```

```
    int i, j, c=1, d;
```

```
    cin>>d;
```

```
    for (i=1; i<=d; i++)
```

```
    {
```

```
        for (j=1; j<=i; j++)
```

```
        {
```

```
            cout<<c<<" ";
```

```
            c++;
```

```
        }
```

```
        cout<<endl;
```

```
    }
```

```
    return 0;
```

```
}
```

```
7
1
2 3
4 5 6
7 8 9 10
11 12 13 14 15
16 17 18 19 20 21
22 23 24 25 26 27 28
```

3.4-5

#include<iostream>// 本题选自《CCF 中学生计算机程序设计》4.24

using namespace std;

```
int main()
{
    int i, j;
    for (i=1; i<=5; i++)
    {
        j=5;
        while (i<=j)
        {
            cout<<i*10+j<<" ";
            j--;
        }
        cout<<endl;
    }
    return 0;
}
```

```
15 14 13 12 11
25 24 23 22
35 34 33
45 44
55
```

3.4-6

#include<iostream>

using namespace std;

```
int main()
{
    int i, j;
    for (i=0, j=10; i<j; i++, j--)
    {
        cout<<i<<" "<<j<<endl;
    }
    return 0;
}
```

```
0 10
1 9
2 8
3 7
4 6
```

3.4-7

```
#include<iostream>
using namespace std;
int main()
{
    int i, j, score=0, sum=0;//score 个数, sum 数字之和
    for (i=1; i<=3; i++)
    {
        for (j=1; j<=i; j++)
        {
            score++;
            sum+=i;
            cout<<i<<" ";
        }
        cout<<endl;
    }
    cout<<" 数字个数 : "<<score<<endl;
    cout<<" 数字之和 : "<<sum;
    return 0;
}
```

```
1
2 2
3 3 3
数字个数 :6
数字之和 :14
```

3.4-8

```
#include<iostream>// 九九乘法表
```

```
using namespace std;
```

```
int main()
```

```
{
```

```
    int i, j;
```

```
    for (i=1; i<=9; i++)
```

```
    {
```

```
        for (j=1; j<=i; j++)
```

```
        {
```

```
            cout<<j<<"*"<<i<<"="<<j*i<<" ";
```

```
        }
```

```
        cout<<endl;
```

```
    }
```

```
    return 0;
```

```
}
```

```
1*1=1
1*2=2 2*2=4
1*3=3 2*3=6 3*3=9
1*4=4 2*4=8 3*4=12 4*4=16
1*5=5 2*5=10 3*5=15 4*5=20 5*5=25
1*6=6 2*6=12 3*6=18 4*6=24 5*6=30 6*6=36
1*7=7 2*7=14 3*7=21 4*7=28 5*7=35 6*7=42 7*7=49
1*8=8 2*8=16 3*8=24 4*8=32 5*8=40 6*8=48 7*8=56 8*8=64
1*9=9 2*9=18 3*9=27 4*9=36 5*9=45 6*9=54 7*9=63 8*9=72 9*9=81
```

3.4-9

```
#include<iostream>
```

```
using namespace std;
```

```
int main()
```

```
{
```

```
    int i, j, a, c=1;
```

```
    cin>>a;
```

```
    for (i=1; i<=a; i++)
```

```
    {
```

```
        for (j=1; j<=c; j++)
```

```
        {
```

```
            cout<<"*";
```

```
        }
```

```
        c=c+2;
```

```
        cout<<endl;
```

```
    }
```

```
    return 0;
```

```
}
```

```
输入 :3
```

```
输出:
```

```
*
```

```
***
```

```
*****
```

3.4-10

```
#include<iostream>
using namespace std;
int main()
{
    int i, j, a, d, c=1;
    cin>>a;
    d=a;

    for (i=1; i<=a; i++)
    {
        for (j=1; j<=d; j++)
        {
            cout<<" ";
        }
        d--;

        for (j=1; j<=c; j++)
        {
            cout<<"*";
        }
        c=c+2;

        cout<<endl;
    }
    return 0;
}
```

输入 :5

输出:

```
 *
***
*****
*****
*****
```

此段代码产生空格

此段代码产生 *

3.4-11

```
#include<iostream>
using namespace std;
int main()
{
    int i, j, a, d, e=1, c=1;
    cin>>a;
    d=a;

    for (i=1; i<=a; i++)
    {
        for (j=1; j<=d; j++)
        {
            cout<<" ";
        }
        d--;

        for (j=1; j<=e; j++)
        {
            cout<<c;
            c++;
        }
        e=e+2;
        cout<<endl;
    }
    return 0;
}
```

输入: 3

输出:

```
1
234
56789
```

3.4-12

```
#include<iostream>
using namespace std;
int main()
{
    int i, j, a, d, h, e=1;
    cin>>a;
    d=a;
    h=a;
    for (i=1; i<=a; i++)
    {
        for (j=1; j<=d; j++)
            cout<<" ";
        d--;

        for (j=1; j<=e; j++)
            cout<<j;
        e=e+2;

        cout<<endl;
    }

    for (i=1; i<=a-1; i++)
    {
        for (j=1; j<=i+1; j++)
            cout<<" ";

        for (j=1; j<=h; j++)
            cout<<j;
        h=h-2;

        cout<<endl;
    }
    return 0;
}
```

输入: 3

输出:

```
1
123
12345
123
1
```

此段代码产生上半部分数字和空格

此段代码产生下半部分数字和空格

3.4-13

```
#include<iostream>
#include<cstdio>
using namespace std;
int main()
{
    int s,kg;//kg 空格
    char a, i, j, j2;
    s=0;
    cin>>a;

    for (i=a;i>='A';i--)
    {
        for (kg=1;kg<=s;kg++)// 产生空格
        {
            cout<<" ";
        }
        s++;

        for (j=i;j>='A';j--)// 产生前半字母
        {
            cout<<j;
        }

        for (j2='A';j2<=i-1;j2++)// 产生后半字母
        {
            cout<<j2;
        }

        cout<<endl;
    }
    return 0;
}
```

输入: D

输出:

DCBAABC

CBAAB

BAA

A

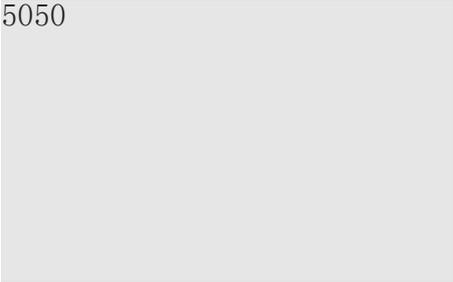
循环练习题

高斯 7 岁那年开始上学。10 岁的时候，一次一位老师想治一治班上的淘气学生，他出了一道数学题，让学生从 $1 + 2 + 3 + \dots$ 一直加到 100 为止。他想这道题足够这帮学生算半天的，他也可能得到半天悠闲。谁知，出乎他的意料，刚刚过了一会儿。小高斯就举起手来，说他算完了。老师一看答案，5050，完全正确，老师惊诧不已。问小高斯是怎么算出来的，高斯说，他不是从开始加到末尾，而是先把 1 和 100 相加，得到 101，再把 2 和 99 相加，也得 101，最后 50 和 51 相加，也得 101，这样一共有 50 个 101，结果当然就是 5050 了，聪明的高斯受到了老师的表扬。

3.5-1

```
#include<iostream>
using namespace std;
int main()
{
    int i, sum;
    sum=0;
    for (i=1; i<=100; i++)
    {
        sum+=i;
    }
    cout<<sum;
    return 0;
}
```

5050



相传，古印度的舍罕王打算重赏国际象棋的发明者——宰相西萨·班·达依尔。于是，这位宰相跪在国王面前说：“陛下，请您在这张棋盘的第一个小格内，赏给我一粒麦子；在第二个小格内给两粒，第三格内给四粒，照这样下去，每一小格都比前一小格加一倍。陛下啊，把这样摆满棋盘上所有 64 格的麦粒，都赏给您的仆人罢！”国王慷慨地答应了宰相的要求，他下令将一袋麦子拿到宝座前。计数麦粒的工作开始了。第一格内放一粒，第二格两粒，第三格四粒……还没到第二十格，袋子已经空了。一袋又一袋的麦子被扛到国王面前来，但是，麦粒数一格接一格地增长得那么迅速，很快就可以看出，即使拿来全印度的小麦，国王也无法兑现他对宰相许下的诺言！这位聪明的宰相到底要求的是多少麦粒呢？

3.5-2

```
#include<iostream>
```

```
using namespace std;
```

```
int main()
```

```
{
```

```
    int n, i;
```

```
    unsigned __int64 mai;//mai 麦子粒
```

// 如果 mai 设置成 int、long long，数值范围太小不能计算；设置 double 计算结果不精准。

```
    cin>>n;
```

```
    mai=1;
```

```
    for (i=2; i<=n; i=i+1)//i 第几天
```

```
    {
```

```
        mai=mai*2;
```

```
        cout<<" 第 "<<i<<" 格子 "<<mai<<endl;
```

```
    }
```

```
    return 0;
```

```
}
```

本例题，如果用 double，因为数据太大，从第 21 格子，使用科学计数法。

科学记数法是一种记数的方法。把一个数表示成 a 与 10 的 n 次幂相乘的形式 ($1 \leq |a| < 10$, a 不为分数形式, n 为整数)，这种记数法叫做科学记数法。

例如：19971400000000=1.99714×10¹³，计算器或电脑表达 10 的幂是一般是用 E 或 e，也就是 1.99714E13=19971400000000。

64

第 2 格子 2

第 3 格子 4

第 15 格子 16384

第 23 格子 4194304

第 35 格子 17179869184

第 51 格子 1125899906842624

第 52 格子 2251799813685248

第 64 格子 9223372036854775808

64

第 2 格子 2

第 3 格子 4

第 4 格子 8

.....

第 20 格子 524288

第 21 格子 1.04858e+006

第 22 格子 2.09715e+006

.....

第 63 格子 4.61169e+018

第 64 格子 9.22337e+018

在一个荷花池中，荷花第一天开放的只是一小部分，第二天，它们开放的数量会是已开放的两倍，之后的每一天，荷花都会以前一天两倍的数量开放……到了第30天，开满了池塘。请问：在第几天荷花开了一半？第15天吗？错！是第29天。这就是著名的荷花定律。

假设，第一天开了1朵，连开30天，计算下每天开了多少朵荷花？

3.5-3

```
#include<iostream>
using namespace std;
int main()
{
    int he, i;
    he=1; // 第一天开的荷花
    for (i=1; i<=30; i++)
    {
        cout<<"第 "<<i<<" 天: "<<he<<endl;
        he=he*2;
    }
    return 0;
}
```

```
第 1 天: 1
第 2 天: 2
第 3 天: 4
第 4 天: 8
第 5 天: 16
第 6 天: 32
第 7 天: 64
第 8 天: 128
第 9 天: 256
第 10 天: 512
第 11 天: 1024
第 12 天: 2048
第 13 天: 4096
第 14 天: 8192
第 15 天: 16384
第 16 天: 32768
第 17 天: 65536
第 18 天: 131072
第 19 天: 262144
第 20 天: 524288
第 21 天: 1048576
第 22 天: 2097152
第 23 天: 4194304
第 24 天: 8388608
第 25 天: 16777216
第 26 天: 33554432
第 27 天: 67108864
第 28 天: 134217728
第 29 天: 268435456
第 30 天: 536870912
```

竹子定律是指，竹子用4年时间生长，竹芽只能长3厘米，而且这3厘米还是深埋于土下。到了第5年，竹子终于能破土而出，以每天30厘米的速度疯长，仅用很短时间就能长到15米。

本题不适合用 for，因为不知道用了多少天才长到15米。用 while 比较合适，我们可以设定长到15米程序就停止运行。

3.5-4

```
#include<iostream>
using namespace std;
int main()
{
    double height;
    int day;
    day=0;// 竹子疯狂生长天数，现在还没开始
    height=0;// 竹子高度，四年没长到地面
    while (height<=15)// 竹子长到15米才停止疯狂生长
    {
        height=height+0.3;// 竹子高度每天增加0.3
        day=day+1;// 增加天数
        cout<<"第"<<day<<"天:"<<height<<endl;
    }
    return 0;
}
```

```
第1天:0.3
第2天:0.6
第3天:0.9
第4天:1.2
第5天:1.5
第6天:1.8
第7天:2.1
第8天:2.4
第9天:2.7
第10天:3
第11天:3.3
第12天:3.6
第13天:3.9
第14天:4.2
第15天:4.5
第16天:4.8
第17天:5.1
第18天:5.4
第19天:5.7
第20天:6
第21天:6.3
第22天:6.6
第23天:6.9
第24天:7.2
第25天:7.5
第26天:7.8
第27天:8.1
第28天:8.4
第29天:8.7
第30天:9
第31天:9.3
第32天:9.6
第33天:9.9
第34天:10.2
第35天:10.5
第36天:10.8
第37天:11.1
第38天:11.4
第39天:11.7
第40天:12
第41天:12.3
第42天:12.6
第43天:12.9
第44天:13.2
第45天:13.5
第46天:13.8
第47天:14.1
第48天:14.4
第49天:14.7
第50天:15
```

国王将金币作为工资，发放给忠诚的骑士。第1天，骑士收到一枚金币；之后2天（第2、3天）里，每天收到2枚金币；之后3天（第4、5、6天）每天收到3枚金币；之后4天（第7、8、9、10天）每天收到4枚金币……这种工资发放模式会一直这样延续下去。当连续N天每天收到N枚金币后，骑士会在之后的连续N+1天里，每天收到N+1枚金币。

已知N为365，请计算从第一天开始的给定天数内，骑士一共获得多少金币？

根据题意，国王发放金币数的规律为1, 2, 2, 3, 3, 3, 4, 4, 4, 4, ……使用双重循环结构按照此规律列举每天的金币数量并累计，直到发放365次后结束循环。

3.5-5-a

```
#include<iostream>
using namespace std;
int main()
{
    int i, j, score=0, sum=0; //score 统计天数, sum 统计金币之和

    for (i=1; i<=365; i++) //365 是估计填写的, 增大或者减小该数字, 观察变化
    {
        for (j=1; j<=i; j++) //作用是每行产生几个数字
        {
            score++; //计算产生几个数字, 也是多少天
            sum+=i; //发金币总数量
            cout<<i<<" ";
            if (score==365)
                break; //到达 365, 停止 j 循环 (但是不能停止 i 循环)
        }
        cout<<endl; //换行
    }
    cout<<" 天数: " <<score<<endl;
    cout<<" 金币总价值: " <<sum;
    return 0;
}
```

另外一种终止程序运行写法:

```
if (score==365)
{
    cout<<endl<<" 天数: " <<score<<endl;
    cout<<" 金币总价值: " <<sum;
    return 0;
} //改写代码, 成功跳出循环, 终止运行。
}
```

```

1
2 2
3 3 3
4 4 4 4
5 5 5 5 5
6 6 6 6 6 6
7 7 7 7 7 7 7
8 8 8 8 8 8 8 8
9 9 9 9 9 9 9 9 9
10 10 10 10 10 10 10 10 10 10
11 11 11 11 11 11 11 11 11 11 11
12 12 12 12 12 12 12 12 12 12 12 12
13 13 13 13 13 13 13 13 13 13 13 13 13
14 14 14 14 14 14 14 14 14 14 14 14 14 14
15 15 15 15 15 15 15 15 15 15 15 15 15 15 15
16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16
17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17
18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18
19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
21 21 21 21 21 21 21 21 21 21 21 21 21 21 21 21 21 21 21 21 21 21
22 22 22 22 22 22 22 22 22 22 22 22 22 22 22 22 22 22 22 22 22 22
23 23 23 23 23 23 23 23 23 23 23 23 23 23 23 23 23 23 23 23 23 23 23
24 24 24 24 24 24 24 24 24 24 24 24 24 24 24 24 24 24 24 24 24 24 24 24
25 25 25 25 25 25 25 25 25 25 25 25 25 25 25 25 25 25 25 25 25 25 25 25 25
26 26 26 26 26 26 26 26 26 26 26 26 26 26 26 26 26 26 26 26 26 26 26 26 26 26
27 27 27 27 27 27 27 27 27 27 27 27 27 27 27 27 27 27 27 27 27 27 27 27 27 27
天数: 365
金币总价值: 6579

```

3. 5-5-b

`#include<iostream>`// 输入天数，求出金币数 推荐学习

```

using namespace std;

int main()
{
    int n, day=0, sum=0;//day 天数, sum 金币之和
    cin>>n;
    for (int i=1;i<=n;i++)
    {
        for (int j=1;j<=i;j++)
        {
            day++;
            sum+=i;
            if (day==n)
            {
                cout<<sum;
                return 0;
            }
        }
    }
}

```

3.5-5-c

```
#include<iostream>// 国王发金币
```

```
using namespace std;
```

```
int main()
```

```
{  
    int i=1, j, score=0, sum=0;//score 统计多少天, sum 统计数字之和  
    while(score<365)  
    {  
        for(j=1;j<=i;j++)  
        {  
            score++;// 计算产生多少数字 , 也是多少天  
            sum+=i; // 发金币总价值  
            cout<<i<<" ";  
            if(score==365)  
                break;  
        }  
        i++;  
        cout<<endl;  
    }  
    cout<<" 天数: "<<score<<endl;  
    cout<<" 金币之和: "<<sum;  
    return 0;  
}
```

穷举

鸡兔同笼，共有 35 头，94 足，问鸡兔各几只？

3.6-1

```
#include<iostream>
```

```
using namespace std;
```

```
int main()
```

```
{
```

```
    int ji, tu;
```

```
    for (ji=0;ji<=35;ji++)
```

```
    {
```

```
        for (tu=0;tu<=35;tu++)
```

```
        {
```

```
            if (ji+tu==35&&ji*2+tu*4==94)
```

```
            {
```

```
                cout<<" 鸡: "<<ji<<" 兔: "<<tu<<endl;
```

```
            }
```

```
        }
```

```
    }
```

```
    return 0;
```

```
}
```

鸡: 23 兔: 12

3.6-2

```
#include<iostream>
using namespace std;
int main()
{
    int x, a, b, c;
    for (x=100; x<=999; x++)
    {
        a=x/100;
        b=x/10%10;
        c=x%10;
        if (a*a*a+b*b*b+c*c*c==x)
        {
            cout<<x<<endl;
        }
    }
    return 0;
}
```

水仙花，例如

$$407=4^3+0^3+7^3$$

```
153
370
371
407
```

3.6-3

```
#include <bits/stdc++.h> // 雷劈数
using namespace std;
int main()
{
    int x, y;
    for (int i=1000; i<=9999; i++)
    {
        x = i/100;
        y = i%100;
        if ((x+y)*(x+y)==i)
        {
            cout<<i<<endl;
        }
    }
    return 0;
}
```

```
2025
3025
9801
```

100 元买 100 只鸡，公鸡每只 5 元，母鸡每只 3 元，小鸡 3 只 1 元。有几种买法？

3.6-4

```
#include<iostream>
```

```
using namespace std;
```

```
int main()
```

```
{
```

```
    int x, y, z;
```

```
    for (x=0;x<=100/5;x++) // 公鸡
```

```
    {
```

```
        for (y=0;y<=100/3;y++) // 母鸡
```

```
        {
```

```
            for (z=0;z<=3*100;z++) // 小鸡
```

```
            {
```

```
                if (5*x+3*y+z/3==100&& x+y+z==100)
```

```
                {
```

```
                    cout<<" 公鸡: "<<x<<" 母鸡: "<<y<<" 小鸡: "<<z<<endl;
```

```
                }
```

```
            }
```

```
        }
```

```
    }
```

```
    return 0;
```

```
}
```

公鸡: 0	母鸡: 25	小鸡: 75
公鸡: 3	母鸡: 20	小鸡: 77
公鸡: 4	母鸡: 18	小鸡: 78
公鸡: 7	母鸡: 13	小鸡: 80
公鸡: 8	母鸡: 11	小鸡: 81
公鸡: 11	母鸡: 6	小鸡: 83
公鸡: 12	母鸡: 4	小鸡: 84

连续输入

3.7-1

`#include<iostream>`// 连续输入：例如收款

```
using namespace std;
int main()
{
    int a,b;
    while(a!=0&&b!=0)
    {
        cin>>a>>b;
        cout<<a+b<<endl;
    }
    return 0;
}
```

连续输入大于 0 的整数，正常运行；输入 0，退出运行。

3.7-2

`#include<iostream>`// 求最大值，本题选自《CCF 中学生计算机程序设计》4.7

```
using namespace std;
int main()
{
    int i,n;
    float x,max=0;//max 最大身高
    cin>>n;
    for(i=1;i<=n;i++)
    {
        cin>>x;
        if(x>max)
            max=x;
    }
    cout<<max<<endl;
    return 0;
}
```

输入：

3

1.6 1.7 1.8

输出：

1.8

求最大数和最小数（程序文件名为 AA.PAS，20 分）

已知 n 个整数序列 $X_1, X_2, X_3, \dots, X_n$ ($n \leq 1000$)，要求找出其中的最大数和最小数。

输入： $n, X_1, X_2, X_3, \dots, X_n$

输出： 第一行显示最大数

第二行显示最小数

样例： 键盘输入： 4 12 3 7 9

屏幕显示： 12

3

3.7-3

`#include<iostream>`// 本题选自 2005 年青岛市程序设计竞赛试题（小学组）

`using namespace std;`// 求最大值和最小值

`int main()`

`{`

`int i, n;`// INT_MIN 是最小的整数， INT_MAX 是最大的整数。

`float x, max=INT_MIN, min=INT_MAX;`//max 最大值， min 最小值。

`cin>>n;`

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

`{`

`cin>>x;`

`if (x>max)`

`max=x;`

`if (x<min)`

`min=x;`

`}`

`cout<<" 最大值: "<<max<<endl;`

`cout<<" 最小值: "<<min;`

`return 0;`

`}`

输入两个数字，分别是学校上课时间和课外辅导时间，如果加起来超过 8，津津不高兴，输出学习时间最多的一天。

3.7-4

```
#include<iostream>
using namespace std;
int main()
{
    int s1, s2; // 分别表示上学时间, 课后学习时间
    int day=0; // day 表示不高兴的时间
    int max=8; // 表示最长时间 8
    int i;
    for (i=1; i<=7; i++)
    {
        cin>>s1>>s2;
        if (s1+s2>max)
        {
            max=s1+s2;
            day=i;
        }
    }

    if (max>8)
        cout<<day;

    else
        cout<<"0";

    return 0;
}
```

```
5 3
6 2
7 2
5 3
5 4
0 4
0 6
3
```

3.7-5

#include<iostream>// 本题选自《ccf 中学生计算机程序设计》 4.12

```
using namespace std;
int main()
{
    int mima=123456;// 预设密码
    int x=0,n=0; //x 输入密码, n 统计输入次数
    while (n<3&&x!=mima)
    {
        n++;
        cin>>x;
        if(x!=mima)
        {
            cout<<" 错误 "<<endl;// 提示错误
        }
    }
    if(x==mima)
    {
        cout<<" 正确 "<<endl;// 提示正确
    }
    else if(n==3)
    {
        cout<<" 冻结 "<<endl;// 提示冻结
    }

    return 0;
}
```

公约数、公倍数

```
#include<iostream>//3.8-1 求最小公倍数
using namespace std;
int main() {
    int a,b,i,max;//max 最大数
    cin>>a>>b;
    max=a<b?b:a;// 如果 a<b, max=b, 反之 max=a
    for (i=max; i<=a*b; i++)
    {
        cout<<i<<" ";// 观察程序运行
        if (i%a==0&& i%b==0)
        {
            cout<<" 公倍数 " <<i<<endl;
            break;// 终止循环
        }
    }
    return 0;
}
```

```
12 8
12 13 14 15 16 17 18
19 20 21 22 23 24
公倍数 24
```

```
#include<iostream>//3.8-2 求最小公倍数
using namespace std;
int main() {
    int a,b,max;
    cin>>a>>b;
    max=a<b?b:a;// 如果 a<b, i=b, 否则 i=a
    while (max%a!=0 || max%b!=0)
    {
        cout<<max<<" ";// 此句用于观察程序运行
        max++;
    }
    cout<<" 最小公倍数: " <<max;
    return 0;
}
```

```
12 8
12 13 14 15 16 17 18
19 20 21 22 23
最小公倍数: 24
```

3.8-3

```
#include<iostream>// 求最大公约数
```

```
using namespace std;
```

```
int main() {
    int a, b, i, min;//min 最小数
    cin>>a>>b;
    if(a>b)min=b;
    else min=a;
    for (i=min;i>=1;i--)
    {
        cout<<a<<" "<<b<<" 公约数可能是 "<<i<<endl;
        if (a%i==0&&b%i==0)
        {
            cout<<a<<" "<<b<<" 最大公约数 "<<i<<endl;
            break;
        }
    }
    return 0;
}
```

```
12 8
12 8 公约数可能是 8
12 8 公约数可能是 7
12 8 公约数可能是 6
12 8 公约数可能是 5
12 8 公约数可能是 4
12 8 最大公约数 4
```

```
13 17
13 17 公约数可能是 13
13 17 公约数可能是 12
13 17 公约数可能是 11
13 17 公约数可能是 10
13 17 公约数可能是 9
13 17 公约数可能是 8
13 17 公约数可能是 7
13 17 公约数可能是 6
13 17 公约数可能是 5
13 17 公约数可能是 4
13 17 公约数可能是 3
13 17 公约数可能是 2
13 17 公约数可能是 1
13 17 最大公约数 1
```

3.8-4

```
#include<iostream> // 求最大公约数
```

```
using namespace std;
```

```
int main() {
    int a, b, min;//min 最小数
    cin>>a>>b;
    min=a>b?b:a;// 如果 a>b, min=b ; 如果 a<b, min=a
    cout<<"min="<<min<<endl;// 此句用于观察程序运行
    while (a%min!=0 || b%min!=0)
    {
        cout<<min<<" "; // 此句用于观察程序运行
        min--;
    }
    cout<<" 最大公约数: "<<min;
    return 0;
}
```

```
12 8
min=8
8 7 6 5 最大公约数: 4
```

本例题可以很好的观察程序运行机制，8、7、6、5符合 $a\%c!=0 || b\%c!=0$ 条件，所以能够进入 while 循环；4 不符合，它被除余数为 0，所以不能进入 while 循环，直接跳到最后一句运行，然后程序结束，高效率解题方法。

3.8-5

```
#include<iostream>// 欧几里德求最大公约数
using namespace std;
int main()
{
    int a,b,temp;
    cin>>a>>b;// 假设 a>b
    while (b>0) // 被除数为 0 无法计算
    {
        cout<<a<<"/"<<b<< "="<<a/b<<" 余数 "<<a%b<<endl;
        temp=a%b;
        a=b;
        b=temp;
        cout<<"a"<< "="<<a<<" "<<"b"<< "="<<b<<endl;
    }
    cout<<" 最大公约数 "<<a;
    return 0;
}
```

```
48 26
48/26=1 余数 22
a=26 b=22
26/22=1 余数 4
a=22 b=4
22/4=5 余数 2
a=4 b=2
4/2=2 余数 0
a=2 b=0
最大公约数 2
```

3.8-6// 本例题选自《GCF 中学生计算机程序设计》。

```
#include<iostream>// 欧几里德求最大公约数
using namespace std;
int main()
{
    int a,b,temp;
    cin>>a>>b;// 假设 a>b
    while (temp=a%b) // 如果除法可以执行，运行程序，否则停止。
    {
        cout<<a<<"/"<<b<< "="<<a/b<<" 余数 "<<a%b<<endl;
        a=b;
        b=temp;
        cout<<"a"<< "="<<a<<" "<<"b"<< "="<<b<<endl;
    }
    cout<<" 最大公约数 "<<b;
    return 0;
}
```

```
48 26
48/26=1 余数 22
a=26 b=22
26/22=1 余数 4
a=22 b=4
22/4=5 余数 2
a=4 b=2
最大公约数 2
```

数列

斐波那契数列指的是这样一个数列：1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89...

这个数列从第3项开始，每一项都等于前两项之和。

斐波那契数列的定义者，是意大利数学家莱昂纳多·斐波那契（Leonardo Fibonacci）。斐波那契数列（Fibonacci sequence），又称黄金分割数列，因数学家莱昂纳多·斐波那契（Leonardo Fibonacci）以兔子繁殖为例子而引入，故又称为“兔子数列”。

3.9-1

```
#include<iostream>
using namespace std;
int main()
{
    long long i;
    int a=0, b=1, c;
    cout<<a<<" "<<b<<" ";
    for (i=3; i<=40; i++)
    {
        c=a+b;
        cout<<c<<" ";
        a=b; b=c;
    }
    return 0;
}
```

```
0 1 1 2 3 5 8 13 21
34 55 89 144 233
377 610 987 1597
2584 4181 6765 10946
17711 28657 46368
75025 121393 196418
317811 514229 832040
1346269 2178309
3524578 5702887
9227465 14930352
24157817
39088169 63245986
```

3.9-2

```
#include<iostream>
using namespace std;
int main()
{
    int i, sum;
    sum=0;
    for (i=0; i<=100; i+=5)
    {
        sum+=i;
        cout<<i<<" ";
    }
    cout<<endl<<sum;
    return 0;
}
```

等差数列，就是数列中任意相邻两项的差值都相等，这个差值成为公差。

```
0 5 10 15 20 25 30
35 40 45 50 55 60 65
70 75 80 85 90 95
100
1050
```

3.9-3

```
#include<iostream> // 等比数列
using namespace std;
int main()
{
    int i, a, b=1;
    cin>>a;
    for (i=1; i<=a; i++)
    {
        cout<<b<<" ";
        b=b*2;
    }
    return 0;
}
```

等比数列

```
5
1 2 4 8 16
```

冰雹猜想

1976年的一天,《华盛顿邮报》于头版头条报道了一条数学新闻。文中记叙了这样一个故事:

70年代中期,美国各所名牌大学校园内,人们都像发疯一般,夜以继日,废寝忘食地玩弄一种数学游戏。这个游戏十分简单:任意写出一个正整数 N ,并且按照以下的规律进行变换:如果是个奇数,则下一步变成 $3N+1$ 。

如果是个偶数,则下一步变成 $N/2$ 。

不单单是学生,甚至连教师、研究员、教授与学究都纷纷加入。为什么这种游戏的魅力经久不衰?因为人们发现,无论 N 是怎样一个数字,最终都无法逃脱回到谷底1。准确地说,是无法逃出落入底部的 $4-2-1$ 循环,永远也逃不出这样的宿命。

这就是著名的“冰雹猜想”。

又称角谷猜想,因为是一个名叫角谷的日本人把它传到中国。

3.9-4

```
#include<iostream>
using namespace std;
int main()
{
    int a;
    cin>>a;
    while(a!=1)
    {
        if(a%2==0)
        {
            a=a/2;
        }
        else
        {
            a=a*3+1;
        }
        cout<<a<<" ";
    }
    return 0;
}
```

```
7
22 11 34 17 52 26 13
40 20 10 5 16 8 4 2
1
```

阶乘

3.10

```
#include<iostream>// 阶乘
using namespace std;
int main()
{
    int i, a, cheng=1;//cheng 连续乘积的结果
    cin>>a;
    for (i=a; i>=1; i--)
        cheng=cheng*i;
    cout<<cheng;
    return 0;
}
```

整数幂

3.11// 本例题选自《CCF 中学生计算机程序设计》。

`#include<iostream>`// 整数幂是指几个几相乘，例如 $2*2*2*2$, $3*3*3*3$

`using namespace std;`

`int main()`

`{`

`int n;`

`cin>>n;`

`while (n%2==0)`

`{`

`n=n/2;`

`cout<<n<<endl;`// 观察程序运行

`}`

`cout<<"n="<<n<<endl;`// 观察程序运行

`if (n==1)`

`cout<<" 是 2 的整数幂 ";`

`else`

`cout<<" 不是 2 的整数幂 ";`

`return 0;`

`}`

```
1024
512
256
128
64
32
16
8
4
2
1
n=1
是 2 的整数幂
```

```
1028
514
257
n=257
不是 2 的整数幂
```

```
1023
n=1023
不是 2 的整数幂
```

短除法

3.12-1

```
#include<iostream>// 分解数字
```

```
using namespace std;
```

```
int main() {  
    int n;  
    cin>>n;  
    while(n>0)  
    {  
        cout<<n%10<<" ";  
        n/=10;  
    }  
    return 0;  
}
```

```
237  
7 3 2
```

3.12-2

```
#include<iostream>// 求数字个数
```

```
using namespace std;
```

```
int main() {  
    int n, sum, score;  
    sum=0;// 求数字之和  
    score=0;//score 一般用来记录次数或者个数  
    cin>>n;  
    while(n>0)  
    {  
        cout<<n%10<<" ";  
        score++;// 求数字个数  
        sum=sum+n%10; // 求数字之和  
        n/=10;  
    }  
    cout<<endl<<score<<" 个数字 "<<endl<<" 数字之和: "<<sum;  
    return 0;  
}
```

```
237  
7 3 2  
3 个数字  
数字之和: 12
```

3.12-3

```
#include<iostream>
using namespace std;
int main()
{
    int i, a, b, temp;//temp 临时用的数据，用完就丢
    cin>>a>>b;
    for (i=a; i<=b; i++)
    {
        temp=i;
        while (temp>0)
        {
            cout<<temp%10<<" ";
            temp/=10;//temp=temp/10
        }
        cout<<endl;
    }
    return 0;
}
```

```
25 27
5 2
6 2
7 2
```

3.12-4

```
#include<iostream>
using namespace std;
int main()
{
    int score, sum, i, a, b, temp; //temp 临时用的数据，用完就丢
    score=0; //score 一般用来记录次数或者个数
    sum=0; // 数字的和
    cin>>a>>b;
    for (int i=a; i<=b; i++)
    {
        temp=i;
        while (temp>0)
        {
            score++;
            sum=sum+temp%10;
            cout<<temp%10<<" ";
            temp/=10;
        }
        cout<<endl;
    }
    cout<<" 个数: "<<score<<" 和: "<<sum;
    return 0;
}
```

```
25 27
5 2
6 2
7 2
个数: 6 和: 24
```

本题输入 a,x 两个数字，计算 x 从 8 至 a 出现的次数

3.12-5

```
#include<iostream>
```

```
using namespace std;
```

```
int main()
```

```
{
```

```
    int score, i, a, x, temp; //temp 临时用的数据，用完就丢
```

```
    score=0; //score 一般用来记录分数或者个数
```

```
    cin>>a>>x;
```

```
    for (int i=8; i<=a; i++)
```

```
    {
```

```
        temp=i;
```

```
        while(temp>0)
```

```
        {
```

```
            cout<<temp<<"/"<<10<<" 余数 "<<temp%10<<" ";
```

```
            if(temp%10==x)
```

```
            {
```

```
                score++;
```

```
                cout<<x<<" 出现第 "<<score<<" 次 "<<" ";
```

```
            }
```

```
            temp/=10; //temp=temp/10
```

```
        }
```

```
        cout<<endl;
```

```
    }
```

```
    cout<<score;
```

```
    return 0;
```

```
}
```

```
15    1
```

```
8/10 余数 8
```

```
9/10 余数 9
```

```
10/10 余数 0 1/10 余数 1 1 出现第 1 次
```

```
11/10 余数 1 1 出现第 2 次 1/10 余数 1 1 出现第 3 次
```

```
12/10 余数 2 1/10 余数 1 1 出现第 4 次
```

```
13/10 余数 3 1/10 余数 1 1 出现第 5 次
```

```
14/10 余数 4 1/10 余数 1 1 出现第 6 次
```

```
15/10 余数 5 1/10 余数 1 1 出现第 7 次
```

```
7
```

3.12-6

```
#include<iostream>
using namespace std;
int main()
{
    int score, i, a, b, x, temp; //temp 临时用的数据，用完就丢
    score=0; //score 一般用来记录分数或者个数
    cin>>a>>b>>x;
    for (int i=a; i<=b; i++)
    {
        temp=i;
        while (temp>0)
        {
            if (temp%10==x)
            {
                score++;
            }
            temp/=10; //temp=temp/10
        }
    }
    cout<<x<<" 出现 "<<score<<" 次 ";
    return 0;
}
```

123 133 3

3 出现 6 次

质数

质数又称素数。一个大于1的自然数，除了1和它自身外，不能被其他自然数整除的数叫做质数，否则称为合数（规定1既不是质数也不是合数）。

3.13-1 求质数问题，本段代码为观察程序运行而写

```
#include<iostream>// 输入 7, 8, 33 测试
```

```
using namespace std;
```

```
int main()
```

```
{
```

```
    int b, i;
```

```
    cin>>b;
```

```
    i=2;
```

```
    while (b%i !=0)
```

```
    {
```

```
        cout<<b<<"%"<<i<<" 余数 "<<b%i<<endl;// 观察程序运行
```

```
        i++;
```

```
    }
```

```
    cout<<b<<"%"<<i<<" 余数 "<<b%i<<" 跳过 while 循环 "<<endl;// 观察程序运行
```

```
    if (i==b) cout<<b<<" 确定是质数 ";
```

```
    return 0;
```

```
}
```

输入 7，可以清楚看到程序逐个数字验证的过程。

输入 8，可以看到因为余数为 0 终止运行。

输入 33，在除以 2 的时候难以确定，除以 3 的时候因为余数为 0 终止运行，确定不是质数。

```
7
7%2 余数 1
7%3 余数 1
7%4 余数 3
7%5 余数 2
7%6 余数 1
7%7 余数 0 跳过 while
循环
7 确定是质数
```

```
8
8%2 余数 0
```

```
33
33%2 余数 1
33%3 余数 0
```

3.13-2

#include<iostream>// 求 a-b 之间有几个质数 (a 不能小于等于 1)

using namespace std;

int main()

{

int a,b,i,j;

cin>>a>>b;

for(j=a;j<=b;j++)

{

i=2;

while(j%i!=0)

{

cout<<j<<"%"<<i<<" 余数 "<<j%i<<endl;

i++;

}

if(i==j)

cout<<j<<"_____ 确定是质数 "<<endl<<endl;

}

return 0;

}

```
2 12
2_____ 确定是质数

3%2 余数 1
3_____ 确定是质数

5%2 余数 1
5%3 余数 2
5%4 余数 1
5_____ 确定是质数

7%2 余数 1
7%3 余数 1
7%4 余数 3
7%5 余数 2
7%6 余数 1
7_____ 确定是质数

9%2 余数 1
```

```
11%2 余数 1
11%3 余数 2
11%4 余数 3
11%5 余数 1
11%6 余数 5
11%7 余数 4
11%8 余数 3
11%9 余数 2
11%10 余数 1
11_____ 确定是质数
```

3.13-3-a

```
#include <bits/stdc++.h> // 判断素数效率最高的解法
```

```
using namespace std;
```

```
int main()
```

```
{
```

```
    int n, i;
```

```
    cin>>n;
```

```
    int f = 1;
```

```
    for (i=2; i<=sqrt(n); i++)
```

```
    {
```

```
        if (n%i==0)
```

```
        {
```

```
            f = 0;
```

```
            break;
```

```
        }
```

```
    }
```

```
    if (n<=1)
```

```
    {
```

```
        f = 0;
```

```
    }
```

```
    if (f)        cout<<" 质数 "; //if(f) 完整写法: if(f==1)
```

```
    else        cout<<" 不是质数 ";
```

```
    return 0;
```

```
}
```

3.13-3-b

```
#include <bits/stdc++.h> // 判断素数效率最高的解法
```

```
using namespace std;
```

```
int main()
```

```
{  
    int n, i;  
    cin >> n;  
    bool flag = true;  
    for (i=2; i<=sqrt(n); i++)  
    {  
        if (n%i==0)  
        {  
            flag = false;  
            break;  
        }  
    }  
    if (n<=1)  
    {  
        flag = false;  
    }  
  
    if (flag)  
    {  
        cout << " 质数 ";  
    }  
    else  
    {  
        cout << " 不是质数 ";  
    }  
    return 0;  
}
```

分解质因数

每个合数都可以写成几个质数相乘的形式，其中每个质数都是这个合数的因数，把一个合数用质因数相乘的形式表示出来，叫做分解质因数。如 $30=2\times 3\times 5$ 。分解质因数只针对合数。

3.14-1

```
#include<bits/stdc++.h>
```

```
using namespace std;
```

```
int main()
```

```
{
```

```
    int n, i = 2;        // 从 2 开始找 n 的因子
```

```
    cin>>n;
```

```
    while(n != 1) // 当 n 等于 1 停止运行
```

```
    {
```

```
        if(n % i == 0)        // 如果 i 是 n 的因子
```

```
        {
```

```
            cout<<i<<endl;
```

```
            n = n / i;
```

```
        }
```

```
        else i=i+1; // 看下一个数
```

```
    }
```

```
    return 0;
```

```
}
```

```
30
```

```
2
```

```
3
```

```
5
```

3.14-2

#include<iostream>// 本题选自《ccf 中学生计算机程序设计》

using namespace std;

int main()

{

int a, i;

cin>>a;

i=2;

while(a!=1)// 当 a 等于 1 停止运行

{

if(a%i==0)

{

cout<<a<<"/"<<i<< "="<<a/i<<endl;

a=a/i;

}

i++;

}

return 0;

}

150

150/2=75

75/3=25

25/5=5

5/5=1

36

36/2=18

18/2=9

9/3=3

3/3=1

13

13/13=1

3.14-3

```
#include<iostream>
using namespace std;
int main()
{
    int a, i;
    cin>>a;

    i=2;
    cout<<a<<"=";
    while(a!=1)
    {
        if(a%i==0)
        {
            cout<<i;
            a=a/i;
            if(a!=1)
                cout<<"*";
        }
        i++;
    }
    return 0;
}
```

36

36=2*2*3*3

因子

一个数的因子和，不包括它本身的所有因子之和，如 12 的因子有 1、2、3、4、6 所以 12 的因子和是 16。现在给定一个数 $n(n \leq 10^9)$ ，求它的因子和。

输入格式：一个数，代表被分解的数

输出格式：一个数，代表因子之和

样例输入：12

样例输出：16

数据提示

60% 的数据， $n \leq 10,000$;

80% 的数据， $n \leq 1,000,000$;

100% 的数据， $n \leq 1,000,000,000$;

这题第一次写的时候，一般人都会想到直接枚举，即直接从 1 一直枚举到该数除本身的最大因子。于是有了下面这一段代码：

3.15-1

```
#include <stdio>
```

```
int main()
```

```
{
```

```
    long n, s = 0, x; // long 长整数
```

```
    int i;
```

```
    scanf("%ld", &n); // ld 数据类型 long
```

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

```
    {
```

```
        if (n % i == 0)
```

```
        {
```

```
            printf("%3ld", i); // 场宽为 3
```

```
            s += i;
```

```
        }
```

```
    }
```

```
    printf("\n"); // 换行符
```

```
    printf("%ld", s);
```

```
    return 0;
```

```
}
```

```
12
 1  2  3  4  6
16
```

但是这段代码一遇到大数就超时。原因是从 1 一直枚举到最大因子数量太多，效率太低，数字一大就会超时。这里有一种解决方案，就是当枚举一个因子时，立马将输入的数除以这个因子，得到另一个因子，这样就可以只枚举到 n 的算术平方根了，即复杂度从 n 降到 \sqrt{n} ，代码如下：

3.15-2

```
#include <stdio.h>
#include <math.h>
int main()
{
    long n, s = 0;
    int i;
    scanf("%ld", &n);
    for (i = 1; i <= sqrt(n); i++)
    {
        if (n % i == 0)
        {
            if (i == n / i) // 遇到根号 n 就只加一次
                s += i;
            else
                s = s + i + n / i;
        }
    }
    printf("%ld", s - n); // 思考，这里为什么减去 n？
    return 0;
}
```

排列组合

3.16-1

```
#include<iostream>// 线段
using namespace std;// 组合数学
int main() {
    int n, sum=0;
    cin>>n;
    sum=n*(n-1);
    cout<<sum;
    return 0;
}
```

平面内有 n 个点，
以其中 2 个点为端点的
有向线段共有多少条？

输入：10
输出：90

3.16-2

```
#include<iostream>// 线段
using namespace std;// 组合数学
int main() {
    int n, sum=0;
    cin>>n;
    sum=n*(n-1)/2;
    cout<<sum;
    return 0;
}
```

平面内有 n 个点，
以其中 2 个点为端点的
线段共有多少条？

输入：10
输出：45

3.16-3

```
#include<iostream>// 多边形有几条对角线？
using namespace std;// 组合数学
int main() {
    int n sum=0;
    cin>>n;
    sum=n*(n-1)/2-n;
    cout<<sum;
    return 0;
}
```

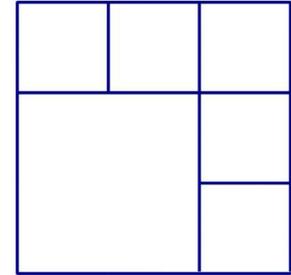
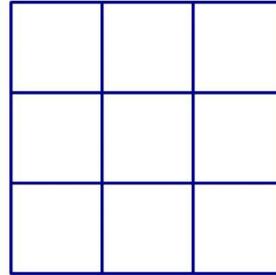
输入：5
输出：5

加法原理、乘法原理

3.17-1

`#include<iostream>`//n*n 正方形包含几个正方形?

```
using namespace std;
int main()
{
    int i, n;
    int sum=0;
    cin>>n;
    for (i=n; i>=1; i--)
    {
        sum+=i*i;
    }
    cout<<sum;
    return 0;
}
```



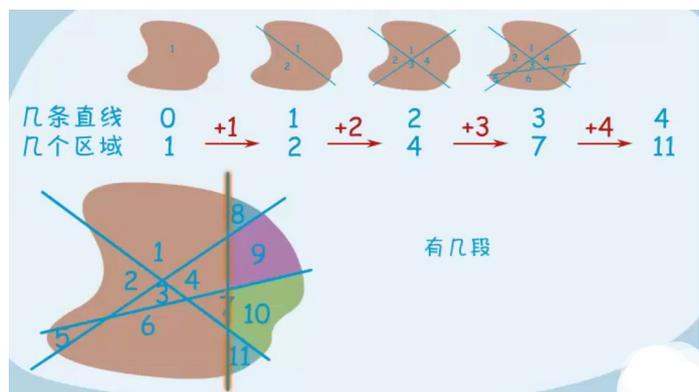
输入: 5
输出: 55

输入: 6
输出: 91

3.17-2

`#include<iostream>`// 直线平分面

```
using namespace std;
int main()
{
    int i, n;
    int sum;//sum 表示图形块的数量
    cin>>n;//n 切分图形的次数
    sum=1;
    for (i=0; i<n; i++)// 是切分图形的第几次
    {
        sum=sum+i;
        cout<<sum<<" ";
    }
    return 0;
}
```



7
1 2 4 7 11 16 22

位运算

3. 18-1

`#include<iostream>`// 输入十进制数字，输出二进制位数

`using namespace std;`

`int main()`

`{`

`int n, num=0;`

`cin>>n;`

`while (n!=0)`

`{`

`num++;`// 统计二进制位数

`n/=2;`

`}`

`cout<<num;`

`return 0;`

`}`

1024

11

3. 18-2

`#include<iostream>`// 输入十进制数字，输出二进制位数 位运算

`using namespace std;`

`int main()`

`{`

`int n, num=0;`

`cin>>n;`

`while (n!=0)`

`{`

`num++;`

`n>>=1;`// 右移，位运算

`}`

`cout<<num;`

`return 0;`

`}`

1024

11

异或运算

输入两个数字，分别是笔试成绩和上机成绩，输出只有一门成绩不合格的学生人数。

3. 19-1

`#include<iostream>`// 选自《CCF 中学生计算机程序设计》 4. 34

```
using namespace std;
```

```
int main()
```

```
{
```

```
    float x, y;
```

```
    int n, i, score=0;
```

```
    cin>>n;
```

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

```
    {
```

```
        cin>>x>>y;
```

```
        if (x<60^y<60) // 异或操作
```

```
            score++;
```

```
    }
```

```
    cout<<score;
```

```
    return 0;
```

```
}
```

```
4
90 0
89 80
90 90
58 76
2
```

学生住宿，A想：我住3床，B住4床；B想：我住1床，D住4床；C想：我住4床，D住3床；D想：我住2床，A住1床。经过老师安排，他们的心愿各自实现一半。

A想：我住3床，B住4床； $(a==3) \wedge (b==4)$

B想：我住1床，D住4床； $(b==1) \wedge (d==4)$

C想：我住4床，D住3床； $(c==4) \wedge (d==3)$

D想：我住2床，A住1床。 $(d==2) \wedge (a==1)$

结果只满足一半，异或操作

3.19-2

`#include<iostream>`// 选自《ccf 中学生计算机程序设计》4.38

`using namespace std;`

`int main()`

`{`

`int a, b, c, d;`

`for (a=1; a<=4; a++)`

`{`

`for (b=1; b<=4; b++)`

`{`

`for (c=1; c<=4; c++)`

`{`

`if (a!=b&&a!=c&&b!=c)`

`{`

`d=1+2+3+4-a-b-c;`

`if ((a==3) ^ (b==4) && (b==1) ^ (d==4) && (c==4) ^ (d==3) && (d==2) ^ (a==1))`

`cout<<a<<b<<c<<d;`

`}`

`}`

`}`

`}`

`return 0;`

`}`

3142