广州大学

程序设计基础

Fundamentals of Programming

主讲人：王国军 教授

计算机科学与网络工程学院

csgjwang@gzhu.edu.cn
http://trust.gzhu.edu.cn/

办公室：行政西楼前座532室

根据周靄如老师《C++程序设计基础》和郑莉老师《C++语言程序设计》制作，仅供广州大学计算机、软件工程、网络工程、网络空间安全、人工智能2023级老师讲课使用。

第2章 程序实例

第2章 C++简单程序设计（一）&（二）

2.2.6 运算符与表达式

2.4 算法的基本控制结构

2.4.1 用if语句实现选择结构

2.4.2 多重选择结构

2.4.3 循环结构

2.4.4 循环结构与选择结构的嵌套

2.4.5 其他控制语句

推荐MOOC视频（见下页）。

第2章 C++简单程序设计（一）

（清华郑莉老师第2章）运算符优先级、类型转换（8分钟）：

https://www.xuetangx.com/learn/THU08091000247/THU08091000247/12424464/video/23270390

优先级	运算符	结合性
1	[] () . -> 后置 ++ 后置 —	左→右
2	前置 ++ 前置 — sizeof & * +(正号) -(负号) ~ !	右→左
3	(强制转换类型)	右→左
4	.* ->*	左→右
5	* / %	左→右
6	+ -	左→右
7	<< >>	左→右
8	< > <= >=	左→右
9	== !=	左→右
10	&	左→右
11	^	左→右
12	|	左→右
13	&&	左→右
14	||	左→右
15	?:	右→左
16	= *= /= %= += -= <<= >>=&= ^= |=	右→左
17	,	左→右

需要的时候查阅一下就行了

需要的时候查阅一下就行了

混合运算时数据类型的转换 —— 隐含转换

- 一些二元运算符（算术运算符、关系运算符、逻辑运算符、位运算符和赋值运算符）要求两个操作数的类型一致
- 在算术运算和关系运算中如果参与运算的操作数类型不一致，编译系统会自动对数据进行转换（即隐含转换），基本原则是将低类型数据转换为高类型数据
- char (unsigned) short (unsigned) int (unsigned) long(unsigned)longlong float double

低 ⟶ 高

条件		转换
有一个操作数是 long double 型。		将另一个操作数转换为 long double 型。
前述条件不满足，并且有一个操作数是 double 型。		将另一个操作数转换为 double 型。
前述条件不满足，并且有一个操作数是 float 型。		将另一个操作数转换为 float 型。
前述条件不满足（两个操作数都不是浮点数）。	有一个操作数是 unsigned long long 型。	将另一个操作数转换为 unsigned long long 型。
	有一个操作数是 long long 型，另一个操作数是 unsigned long 型	两个操作数都转换为 unsigned long long 型。
	前述条件不满足，并且有一个操作数是 unsigned long 型。	将另一个操作数转换为 unsigned long 型。
	前述条件不满足，并且有一个操作数是 long 型，另一个操作数是 unsigned int 型。	将两个操作数都转换为 unsigned long 型。
	前述条件不满足，并且有一个操作数是 long 型。	将另一个操作数转换为 long 型。
	前述条件不满足，并且有一个操作数是 unsigned int 型。	将另一个操作数转换为 unsigned int 型。
	前述条件都不满足	将两个操作数都转换为 int 型。

混合运算时数据类型的转换

- 将一个非布尔类型的算术值赋给布尔类型时，算术值为0则结果为false，否则结果为true
- 将一个布尔值赋给非布尔类型时，布尔值为false则结果为0，布尔值为true则结果为1

混合运算时数据类型的转换

- 将一个浮点数赋给整数类型时，结果值将只保留浮点数中的整数部分，小数部分将丢失
- 将一个整数值赋给浮点类型时，小数部分记为0。如果整数所占的空间超过了浮点类型的容量，精度可能有损失

混合运算时数据类型的转换 —— 显式转换

- 显式类型转换的作用是将表达式的结果类型转换为类型说明符所指定的类型
- 语法形式
  - 类型说明符(表达式)
  - (类型说明符)表达式
  - 类型转换操作符<类型说明符>(表达式)

混合运算时数据类型的转换 —— 显式转换

- 语法形式——C++的形式

  类型转换操作符<类型说明符>(表达式)

- 类型转换操作符可以是：

  const_cast、dynamic_cast、reinterpret_cast、static_cast

- 例：int(z), (int)z, static_cast<int>(z)三种完全等价

C语言运算成分

（8/9）

==========

强制类型转换

强制类型转换

中国大学MOOC

- 形式：
  - (类型名)（表达式）
- 举例：
  - (double)a　　将a的值转换成double类型
  - (int)(x+y)　　将x+y的值转换成int类型
  - (float)(5/3)　　将5/3的值转换成float类型
- 注意：
  - 强制类型转换后，被转换的量的类型并没有发生变化

逻辑运算与逻辑表达式

- “&&”的运算规则
  - 两侧表达式都为真，结果为真
  - 有一侧表达式为假，结果为假
- “&&”的“短路特性” 表达式1 && 表达式2
  - 先求解表达式1
  - 若表达式1的值为false，则最终结果为false，不再求解表达式2
  - 若表达式1的结果为true，则求解表达式2，以表达式2的结果作为最终结果

逻辑运算与逻辑表达式

- “||”的运算规则
  - 两侧表达式都为假，结果为假
  - 有一侧表达式为真，结果为真
- “||”的“短路特性” 表达式1 || 表达式2
  - 先求解表达式1
  - 若表达式1的值为true，则最终结果为true，不再求解表达式2
  - 若表达式1的结果为false，则求解表达式2，以表达式2的结果作为最终结果

C语言运算成分

（7/9）

==========

逻辑运算

之

运算的取舍

思考题


```
#include<iostream>
using namespace std;
int main()
{
   int a = 0, b = 0;
   a = 5 > 3 && 2 || 8 < 4 - (b = !0);
   cout<<a<<" "<<b;
   return 0;
}
```

条件运算（表达式1？表达式2：表达式3）的优先级

- 条件运算符优先级高于赋值运算符，低于逻辑运算符

例如：　x = a > b ? a : b

- 表达式1是bool类型，表达式2、3的类型可以不同，条件表达式的最终类型为 2 和 3 中较高的类型

北大李戈老师中国大学MOOC《计算概论与程序设计基础》

（C语言中的运算成分）赋值运算的说明（11分钟）：

https://www.icourse163.org/learn/PKU-1002529002?tid=1450221458#/learn/content?type=detail&id=1229624123&sm=1

C语言运算成分

（3/9）

=========

赋值运算

附加说明

之

连续赋值表达式

赋值表达式

- 连续的赋值运算
  - 自右而左的结合顺序
    - a = b = 5;　　//a, b均为5
    - a = b = c = 5;　　//a, b, c均为5
      - int a=b=c=5;　　//编译错！
  - a = (b = 4) + (c = 6); //a为10, b为4, c为6
- 举例：

a+=a-=a * a （设a为12）

a=a-a * a （a为12-12 * 12=-132）

a+=-132 → a = a+(-132) → a=-264

北大李戈老师中国大学慕课《计算概论与程序设计基础》

（C语言中的运算成分）自增自减运算（22分钟）：

https://www.icourse163.org/learn/PKU-1002529002?tid=1450221458#/learn/content?type=detail&id=1229624125&sm=1

C语言运算成分

（5/9）

==========

算数运算符

之

自增、自减

算术表达式

- 自增、自减运算符：使变量的值加1或减1
  - ++i，--i
    - 在使用i之前，先将i的值加（减）1
  - i++，i--
    - 在使用i之后，再将i的值加（减）1

C语言运算成分

（5/9）

==========

算数运算符

之

自增、自减

算术表达式

- 自增、自减运算符：使变量的值加1或减1
  - ++i， --i
    - 在使用i之前，先将i的值加（减）1
  - i++， i--
    - 在使用i之后，再将i的值加（减）1
- 例如：i的值为3，则
  - j=++i;
  - j=i++;
  - cout<<++i;
  - cout<<i++;

C语言运算成分

(5/9)

==========

算数运算符

之

自增、自减

自增、自减运算符

- 举例：
  - 设i的值为3；则

```
cout<<-i++<<endl;
cout<<-++i<<endl;
cout<<(-i)++<<endl;
cout<<++i++<<endl;
```

C语言运算成分

(5/9)

==========

算数运算符

之

输出自增自减

自增、自减运算符


```
int main()
{
    int a = 0, b = 0, c = 2, d = 0, e = 2, f = 2;
    cout << a << " " << a++ << " " << endl;
    cout << ++b << " " << b++ << " " << endl;
    cout << c << " " << (c++) + (++c) << " " << endl;
    cout << (d = f++) + (e = f) << endl;
    cout << f << " " << d << " " << e << endl;
    return 0;
}
```

C语言运算成分

(5/9)

==========

算数运算符

之

输出自增自减

自增、自减运算符

```
int main()
{
    int a = 0, b = 0, c = 2, d = 0, e = 2, f = 2;
    cout << a << " " << a++ << " " << endl;
    cout << ++b << " " << b++ << " " << endl;
    cout << c << " " << (c++) + (++c) << " " << endl;
    cout << (d = f++) + (e = f) << endl;
    cout << f << " " << d << " " << e << endl;
    return 0;
}
```

第2章 C++简单程序设计（二）

（清华郑莉老师第2章）选择结构 之 if语句（8分钟）：

https://www.xuetangx.com/learn/THU08091000247/THU08091000247/12424464/video/23270406

If语句的语法形式

- if (表达式) 语句

  例：if (x > y) cout << x;

- if (表达式) 语句1 else 语句2

  例：if (x > y) cout << x;
  else cout << y;

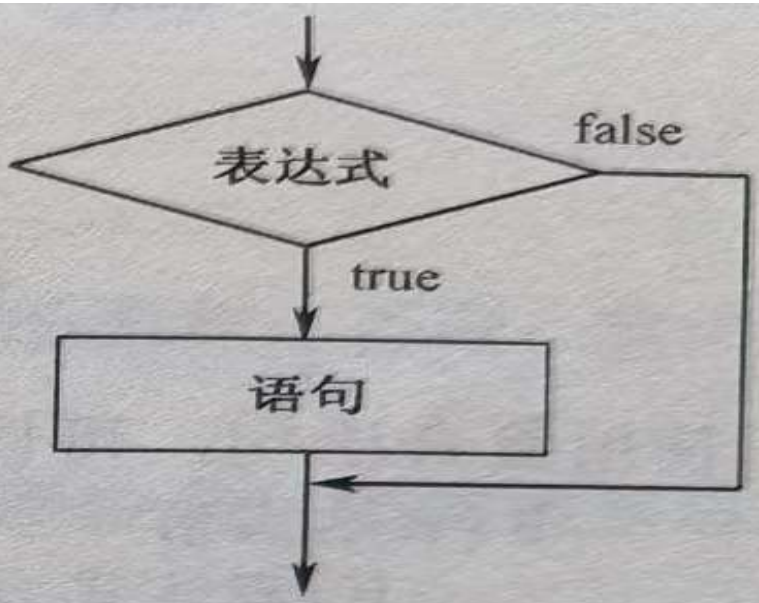


图 2-2　if 语句的执行流程

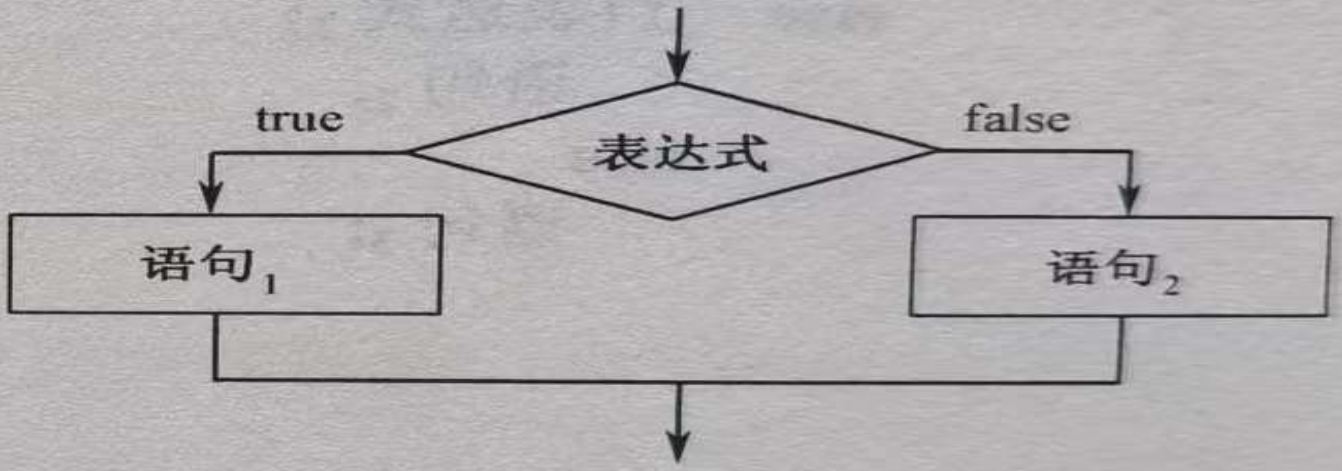


图 2-3　if-else 语句的执行流程

If语句的语法形式

- if (表达式) 语句

  例：if (x > y) cout << x;

- if (表达式) 语句1 else 语句2

  例：if (x > y) cout << x;
  else cout << y;

- if (表达式1) 语句1
  else if (表达式2) 语句2
  else if (表达式3) 语句3
  …
  else 语句 n

If语句的语法形式

```
if (表达式1) 语句1
else
    if (表达式2) 语句2
    else
        if (表达式3) 语句3
            ...
        else 语句 n
```

- if (表达式1) 语句1
  else if (表达式2) 语句2
  else if (表达式3) 语句3
  ...
  else 语句 n

C语言控制成分

(1/3)

==========

分支结构

之

if 语句

```
#include<iostream>
using namespace std;
int main()
{
        float  weight = 0, height = 0, healthRate = 0;
        cin >> weight>>height;
        healthRate = weight / (height*height);
        if ((18 <= healthRate)&&(healthRate <= 25))
                cout << "体重适中！" << endl;
        else if ((25 < healthRate)&&(healthRate <= 30))
                cout << "超重！注意控制！" << endl;
        else if (( 30 < healthRate)&&(healthRate <= 35))
                cout << "肥胖！减肥吧！" << endl;
        else if ((35 < healthRate)&&(healthRate <= 40))
                cout << "重度肥胖！别吃了！" << endl;
        else
                cout << "请直接拨打120！" << endl;
        return 0;
}
```

嵌套的if结构

- 语法形式:

```
if(  )
        if(  ) 语句 1
        else 语句 2
else
        if(  ) 语句 3
        else 语句 4
```

- 注意:
  - 语句 1、2、3、4 可以是复合语句；
  - 每层的 if 与 else 配对，或用 { } 来确定层次关系。

例2-3：输入两个整数，比较两个数的大小。

```
#include<iostream>
using namespace std;
int main() {
   int x, y;
   cout << "Enter x and y:";
   cin >> x >> y;
   if (x != y)
      if (x > y)
         cout << "x > y" << endl;
      else
         cout << "x < y" << endl;
   else
      cout << "x = y" << endl;
   return 0;
}
```

运行结果1：
Enter x and y:5 8
x < y
运行结果2：
Enter x and y:8 8
x = y
运行结果3：
Enter x and y:12 8
x > y

C语言控制成分

(1/3)

==========

分支结构

之

if 语句

```
#include <iostream>
using namespace std;
int main()
{
        int year = 0;
        cin >> year;
        if (year % 4 == 0)
        {
                if (year % 100 == 0)
                {
                        if (year % 400 == 0)
                                cout << "Y";
                        else
                                cout << "N";
                }
                else
                        cout << "Y";
        }
        else
                cout << "N";
        return 0;
}
```

错误写法

```
if( )
        if( ) 语句 1
else
        ...
```

正确写法

```
if( )
{
        if( ) 语句 1
}
else
```

使用复合语句，可以改变条件语句的执行流程。例如，以下形式的语句：

```
if (表达式0)
    if (表达式1)
        语句1;
    else   if (表达式2)
               语句2;
           else
               语句3;
```

等价于

```
if (表达式0)
{   if (表达式1)
        语句1;
    else
    {   if (表达式2)
            语句2;
        else
            语句3;
    }
}
```

第2章 C++简单程序设计（二）

（清华郑莉老师第2章）选择结构 之 switch语句（8分钟）：

https://www.xuetangx.com/learn/THU08091000247/THU08091000247/12424464/video/23270408

例2-4：输入一个0～6的整数，转换成星期


```
#include <iostream>
using namespace std;
int main() {
    int day;
    cin >> day;
    switch (day) {
    case 0: cout << "Sunday" << endl; break;
    case 1: cout << "Monday" << endl; break;
    case 2: cout << "Tuesday" << endl; break;
    case 3: cout << "Wednesday" << endl; break;
    case 4: cout << "Thursday" << endl; break;
    case 5: cout << "Friday" << endl; break;
    case 6: cout << "Saturday" << endl; break;
    default:
      cout<<"Day out of range Sunday .. Saturday"<< endl;   break;
    }
    return 0;
}
```


可以用多重if else语句来解决

switch语句的语法：

◊ 一般形式:

switch语句的语法：

switch （表达式）

{ case 常量表达式 1 : 语句1
case 常量表达式 2 : 语句2
⋮
case 常量表达式 n : 语句n
default : 语句n+1
}

◊ 执行顺序:

- 以case中的常量表达式值为入口标号，由此开始顺序执行。因此，每个case分支最后应该加break语句。

◊ 注意:

- case分支可包含多个语句，且不用{ }。
- 表达式、判断值都是int型或char型。
- 如果若干分支执行内容相同可共用一组语句。

2.2.2 switch语句

if 与 switch 语句互换

逻辑表达式

```
if ( E1)
   S1 ;
else  S2 ;
```

⟺

```
switch ( int (E1) )
{ case  1  :  S1 ;  break ;
   case  0  :  S2 ;
}
```

2.2.2 switch语句

if 与 switch 语句互换

```
if  ( a > b)
   max = a ;
else  max = b ;
```

```
switch ( int (a > b) )
{  case  1  :  max = a ;  break ;
   case  0  :  max = b ;
}
```

2.3 循环控制

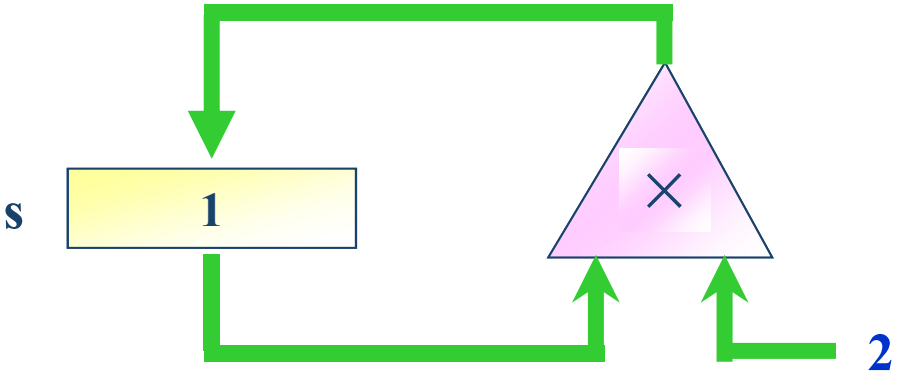
循环概念

为解决某一问题，或求取某一计算结果，特定的条件下，程序中反复地按某一模式进行操作。

问题： 求 2^n

$s = 1;$
$s = s * 2;$
$s = s * 2;$
……
$s = s * 2;$

执行 n 次



2.3 循环控制

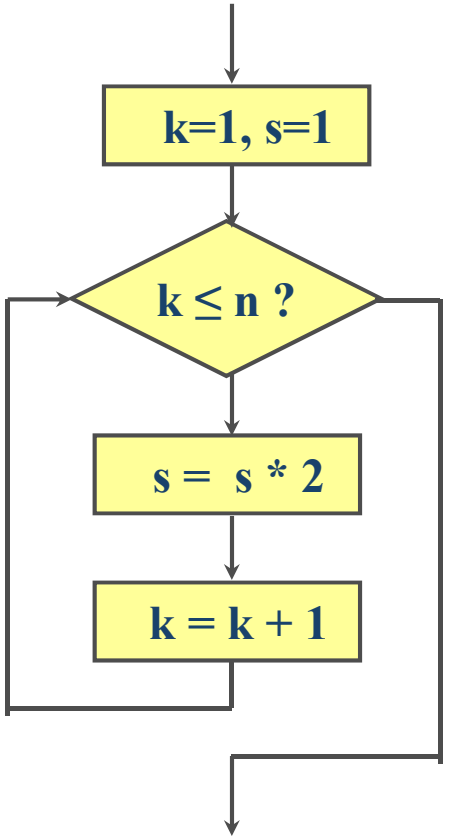
循环概念

为解决某一问题，或求取某一计算结果，特定的条件下，程序中反复地按某一模式进行操作。

问题： 求 2^n

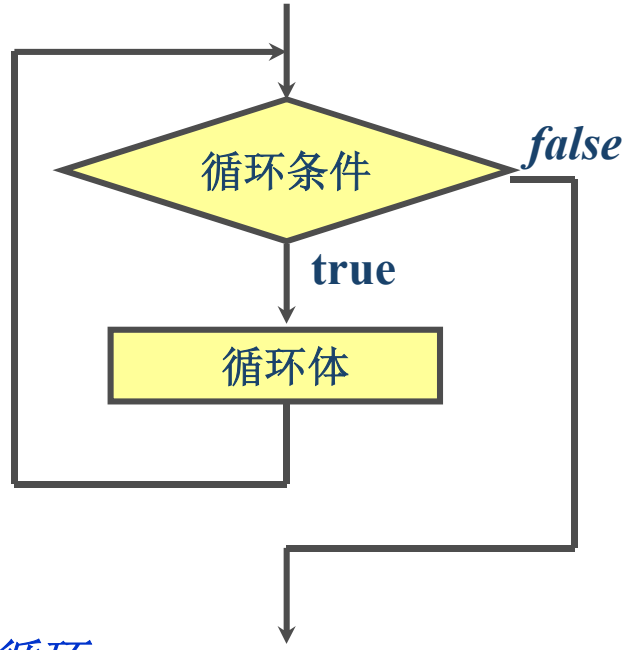
```
s = 1;
s = s * 2;
s = s * 2;
.....
s = s * 2;
```

执行 *n* 次

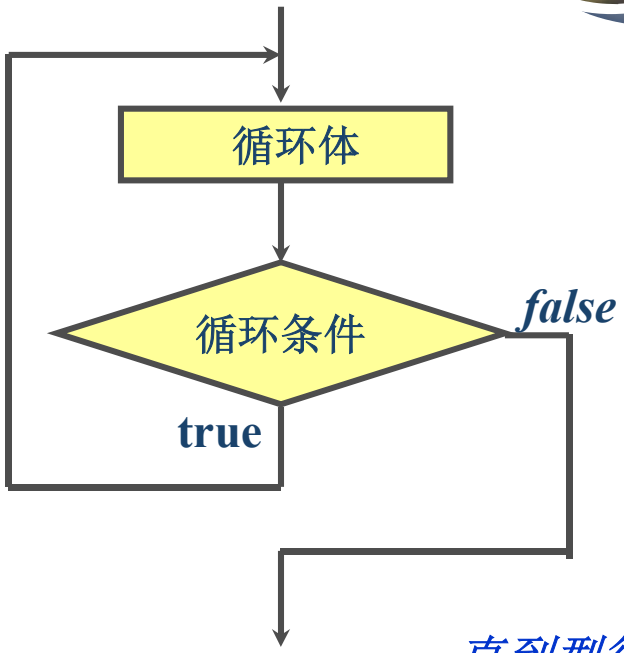


2.3 循环控制

两种典型循环结构



当型循环



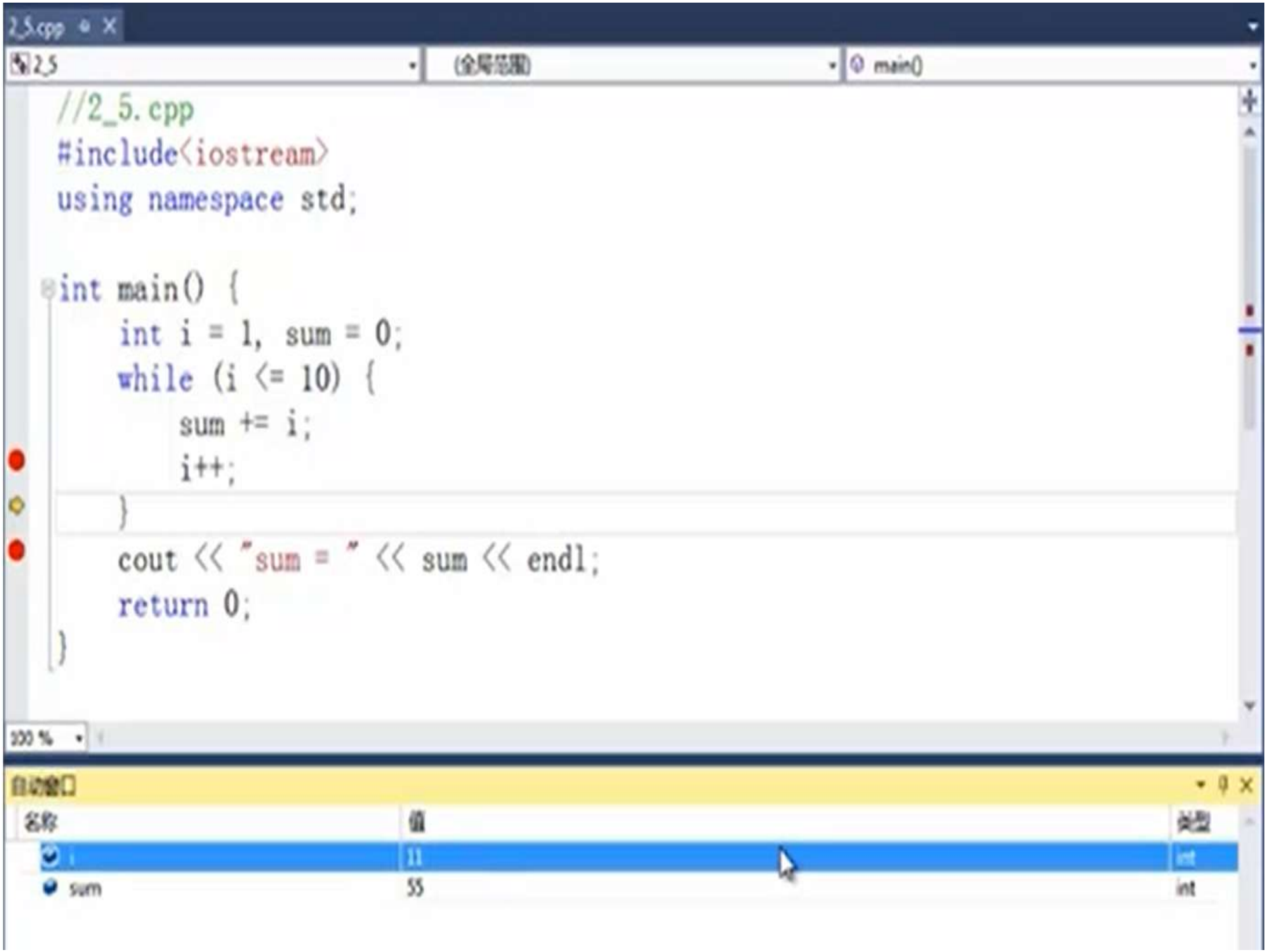
直到型循环

- **循环体的算法是什么?**
- **循环的条件、循环结束条件是什么?**
- **如何修改循环条件?**

第2章 C++简单程序设计（二）

（清华郑莉老师第2章）循环结构 之 while语句（9分钟）：

https://www.xuetangx.com/learn/THU08091000247/THU08091000247/12424464/video/23270414


2_5.cpp
2_5
(全局范围)
main()
//2_5.cpp
#include<iostream>
using namespace std;
int main() {
int i = 1, sum = 0;
while (i <= 10) {
sum += i;
i++;
}
cout << "sum = " << sum << endl;
return 0;
}
100 %
自动窗口
名称
值
类型
i
11
int
sum
55
int

- while语句的语法形式

  while（表达式）语句

  可以是复合语句，其中必须含有改变条件表达式值的语句。

例 2-13 求两个正整数 m 和 n 的最大公约数

辗转相除法：

当 m > n，m 与 n 的最大公约数等于 n 和 m%n 的最大公约数；

当 n = 0，m 和 n 的最大公约数等于 m 。

例如：

m = 24，n = 9

24 和 *9* 的最大公约数等于 (24 % 9) = *6* 和 *9* 的最大公约数；

9 和 *6* 的最大公约数等于 (9 % 6) = *3* 和 *6* 的最大公约数；

6 和 *3* 的最大公约数等于 (6 % 3) = *0* 和 *3* 的最大公约数；

所以， 24 和 9 的最大公约数等于 3。

例 2-13 求两个正整数 m 和 n 的最大公约数

辗转相除法：

当 m > n，m 与 n 的最大公约数等于 n 和 m%n 的最大公约数；

当 n = 0，m 和 n 的最大公约数等于 m 。

```
a = m , b = n , r = b ;
while ( r != 0 )
{ 把 a%b 的值赋给 r ;
  用 b 的值替换 a 的值 ;
  用 r 的值替换 b 的值 ;
}
```

a 是最大公约数

24 % 9

a	b	r
24	9	9

例 2-13 求两个正整数 m 和 n 的最大公约数

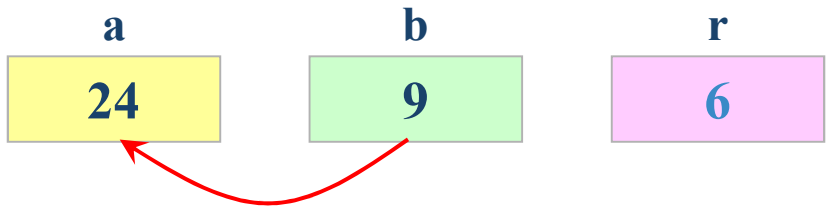
辗转相除法：

当 m > n，m 与 n 的最大公约数等于 n 和 m%n 的最大公约数；

当 n = 0，m 和 n 的最大公约数等于 m 。

```
a = m , b = n , r = b ;
while ( r != 0 )
{ 把 a%b 的值赋给 r ;
  用 b 的值替换 a 的值 ;
  用 r 的值替换 b 的值 ;
}
```

a 是最大公约数



例 2-13 求两个正整数 m 和 n 的最大公约数

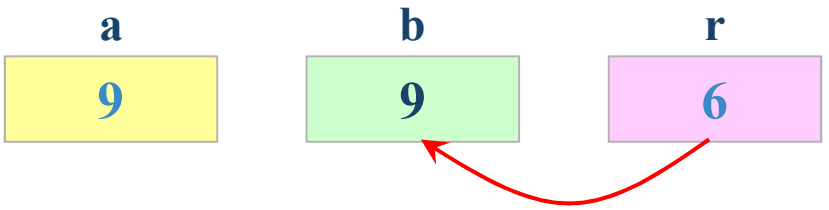
辗转相除法：

当 m > n，m 与 n 的最大公约数等于 n 和 m%n 的最大公约数；

当 n = 0，m 和 n 的最大公约数等于 m 。

```
a = m , b = n , r = b ;
while ( r != 0 )
{ 把 a%b 的值赋给 r ;
  用 b 的值替换 a 的值 ;
  用 r 的值替换 b 的值 ;
}
```

a 是最大公约数



例 2-13 求两个正整数 m 和 n 的最大公约数

辗转相除法：

当 m > n，m 与 n 的最大公约数等于 n 和 m%n 的最大公约数；

当 n = 0，m 和 n 的最大公约数等于 m 。

```
a = m , b = n , r = b ;
while ( r != 0 )
{ 把 a%b 的值赋给 r ;
  用 b 的值替换 a 的值 ;
  用 r 的值替换 b 的值 ;
}
```

a 是最大公约数

9 % 6

a	b	r
9	6	6

例 2-13 求两个正整数 m 和 n 的最大公约数

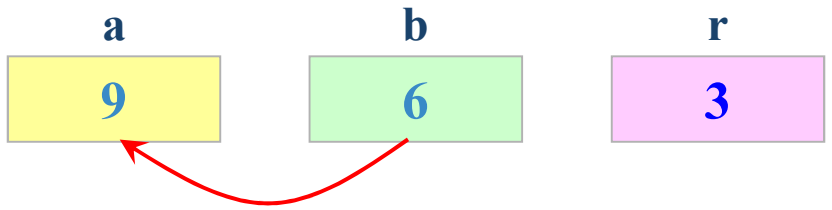
辗转相除法：

当 m > n，m 与 n 的最大公约数等于 n 和 m%n 的最大公约数；

当 n = 0，m 和 n 的最大公约数等于 m 。

```
a = m , b = n , r = b ;
while ( r != 0 )
{ 把 a%b 的值赋给 r ;
  用 b 的值替换 a 的值 ;
  用 r 的值替换 b 的值 ;
}
```

a 是最大公约数



例 2-13 求两个正整数 m 和 n 的最大公约数

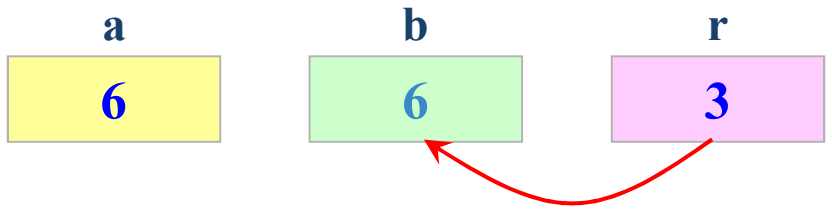
辗转相除法：

当 m > n，m 与 n 的最大公约数等于 n 和 m%n 的最大公约数；

当 n = 0，m 和 n 的最大公约数等于 m 。

```
a = m , b = n , r = b ;
while ( r != 0 )
{ 把 a%b 的值赋给 r ;
  用 b 的值替换 a 的值 ;
  用 r 的值替换 b 的值 ;
}
```

a 是最大公约数



例 2-13 求两个正整数 m 和 n 的最大公约数

辗转相除法：

当 m > n，m 与 n 的最大公约数等于 n 和 m%n 的最大公约数；

当 n = 0，m 和 n 的最大公约数等于 m 。

```
a = m , b = n , r = b ;
while ( r != 0 )
{ 把 a%b 的值赋给 r ;
  用 b 的值替换 a 的值 ;
  用 r 的值替换 b 的值 ;
}
```

a 是最大公约数

6 % 3

a	b	r
6	3	3

例 2-13 求两个正整数 m 和 n 的最大公约数

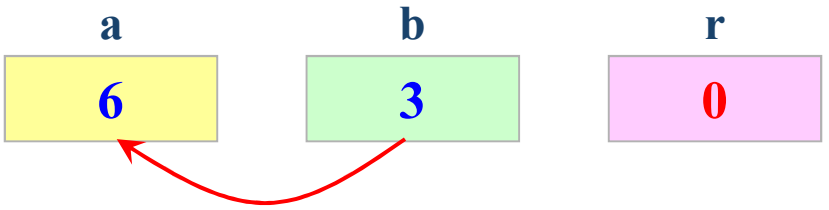
辗转相除法：

当 m > n，m 与 n 的最大公约数等于 n 和 m%n 的最大公约数；

当 n = 0，m 和 n 的最大公约数等于 m 。

```
a = m , b = n , r = b ;
while ( r != 0 )
{ 把 a%b 的值赋给 r ;
  用 b 的值替换 a 的值 ;
  用 r 的值替换 b 的值 ;
}
```

a 是最大公约数



例 2-13 求两个正整数 m 和 n 的最大公约数

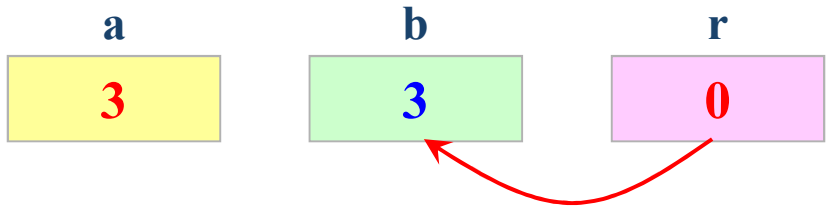
辗转相除法：

当 m > n，m 与 n 的最大公约数等于 n 和 m%n 的最大公约数；

当 n = 0，m 和 n 的最大公约数等于 m 。

```
a = m , b = n , r = b ;
while ( r != 0 )
{ 把 a%b 的值赋给 r ;
  用 b 的值替换 a 的值 ;
  用 r 的值替换 b 的值 ;
}
```

a 是最大公约数



例 2-13 求两个正整数 m 和 n 的最大公约数

辗转相除法：

当 m > n，m 与 n 的最大公约数等于 n 和 m%n 的最大公约数；

当 n = 0，m 和 n 的最大公约数等于 m 。

```
a = m , b = n , r = b ;
while ( r != 0 )
{ 把 a%b 的值赋给 r ;
  用 b 的值替换 a 的值 ;
  用 r 的值替换 b 的值 ;
}
a 是最大公约数
```

a	b	r
3	0	0

3 是 24 和 9 的最大公约数

例 2-13 求两个正整数 m 和 n 的最大公约数

```
using namespace std ;
int main()
{ int m , n , a , b , r ;
  cout << "input two integers :\n" ;
  cout << "? " ;  cin >> m ;
  cout << "? " ;  cin >> n ;
  if ( m > n) { a = m ; b = n ; }
    else { a = n ; b = m ; }
  r = b ;
 while ( r != 0 )
 { r = a % b ;
    a = b ;
    b = r ;
 }
 cout << m << " and " << n << " maximal common divisor is : " << a <<
endl ;
}
```

例 2-13 求两个正整数 m 和 n 的最大公约数

```
using namespace std ;
int main()
{ int m , n , a , b , r ;
  cout << "input two integers :\n" ;        // 提示并输入数据
  cout << "? " ;  cin >> m ;
  cout << "? " ;  cin >> n ;
  if ( m > n) { a = m ; b = n ; }
   else { a = n ; b = m ; }
  r = b ;
 while ( r != 0 )
 { r = a % b ;
    a = b ;
    b = r ;
 }
 cout << m << " and " << n << " maximal common divisor is : " << a <<
endl ;
}
```

例 2-13 求两个正整数 m 和 n 的最大公约数

```
using namespace std ;
int main()
{ int m , n , a , b , r ;
  cout << "input two integers :\n" ;        // 提示并输入数据
  cout << "? " ;  cin >> m ;
  cout << "? " ;  cin >> n ;
  if ( m > n) { a = m ; b = n ; }           // 把大值放在 a 中
    else { a = n ; b = m ; }
  r = b ;
 while ( r != 0 )
 { r = a % b ;
    a = b ;
    b = r ;
 }
 cout << m << " and " << n << " maximal common divisor is : " << a <<
endl ;
}
```

例 2-13 求两个正整数 m 和 n 的最大公约数

```
using namespace std ;
int main()
{ int m , n , a , b , r ;
  cout << "input two integers :\n" ;        // 提示并输入数据
  cout << "? " ;  cin >> m ;
  cout << "? " ;  cin >> n ;
  if ( m > n) { a = m ; b = n ; }           // 把大值放在 a 中
    else { a = n ; b = m ; }
  r = b ;                                   // 余数初值
 while ( r != 0 )
 { r = a % b ;
    a = b ;
    b = r ;
 }
 cout << m << " and " << n << " maximal common divisor is : " << a <<
endl ;
}
```

例 2-13 求两个正整数 m 和 n 的最大公约数

```
using namespace std ;
int main()
{ int m , n , a , b , r ;
  cout << "input two integers :\n" ;        // 提示并输入数据
  cout << "? " ;  cin >> m ;
  cout << "? " ;  cin >> n ;
  if ( m > n) { a = m ; b = n ; }           // 把大值放在 a 中
    else { a = n ; b = m ; }
  r = b ;                                   // 余数初值
 while ( r != 0 )
 { r = a % b ;                              // 求新的余数
    a = b ;
    b = r ;
 }
 cout << m << " and " << n << " maximal common divisor is : " << a <<
endl ;
}
```

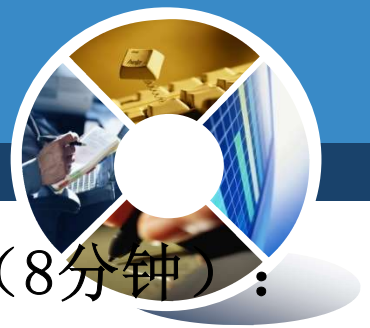
例 2-13 求两个正整数 m 和 n 的最大公约数

```
using namespace std ;
int main()
{ int m , n , a , b , r ;
  cout << "input two integers :\n" ;        // 提示并输入数据
  cout << "? " ;  cin >> m ;
  cout << "? " ;  cin >> n ;
  if ( m > n) { a = m ; b = n ; }           // 把大值放在 a 中
    else { a = n ; b = m ; }
  r = b ;                                   // 余数初值
 while ( r != 0 )
 { r = a % b ;                              // 求新的余数
    a = b ;
    b = r ;                                 // 变量值迭代
 }
 cout << m << " and " << n << " maximal common divisor is : " << a <<
endl ;
}
```

例 2-13 求两个正整数 m 和 n 的最大公约数

```
using namespace std ;
int main()
{ int m , n , a , b , r ;
  cout << "input two integers :\n" ;      // 提示并输入数据
  cout << "? " ;  cin >> m ;
  cout << "? " ;  cin >> n ;
  if ( m > n) { a = m ; b = n ; }          // 把大值放在 a 中
    else { a = n ; b = m ; }
  r = b ;
 while ( r != 0 )
 { r = a % b ;
    a = b ;
    b = r ;
 }
 cout << m << " and " <<
endl ;
}
```

第2章 C++简单程序设计（二）



（清华郑莉老师第2章）循环结构 之 do-while语句（8分钟）：

https://www.xuetangx.com/learn/THU08091000247/THU08091000247/12424464/video/23270416

例2-6：输入一个数，将各位数字翻转后输出

```
#include <iostream>
using namespace std;
int main() {
  int n, right_digit, newnum = 0;
  cout << "Enter the number: ";
  cin >> n;
  cout << "The number in reverse order is  ";
  do {
      right_digit = n % 10;
      cout << right_digit;
      n /= 10;
  } while (n != 0);
   cout << endl;
  return 0;
}
```

运行结果：

```
Enter the number: 365
The number in reverse order is  563
```

然后将输入的数据

- do-while语句的语法形式

  do　语句

  while (表达式)

- 执行顺序

  先执行循环体语句，后判断条件。

  表达式为 true 时，继续执行循环体。

```cpp
//2_7.cpp
#include <iostream>
using namespace std;
int main() {
    int i = 1, sum = 0;
    do {
        sum += i;
        i++;
    } while (i <= 10);
    cout << "sum = " << sum << endl;
    return 0;
}
```

例 2-15 输入二进制数字串，转换成十进制正整数。

二进制数$b_n b_{n-1} \ldots b_2 b_1 b_0$有转换为十进制Dec的公式：

$$Dec = b_n \times 2^n + b_{n-1} \times 2^{n-1} + \ldots + b_2 \times 2^2 + b_1 \times 2^1 + b_0 \times 2^0$$

输入 — 移位 转换

Binary

输入串

0	1	0	1	1	0	0	1

Decimal

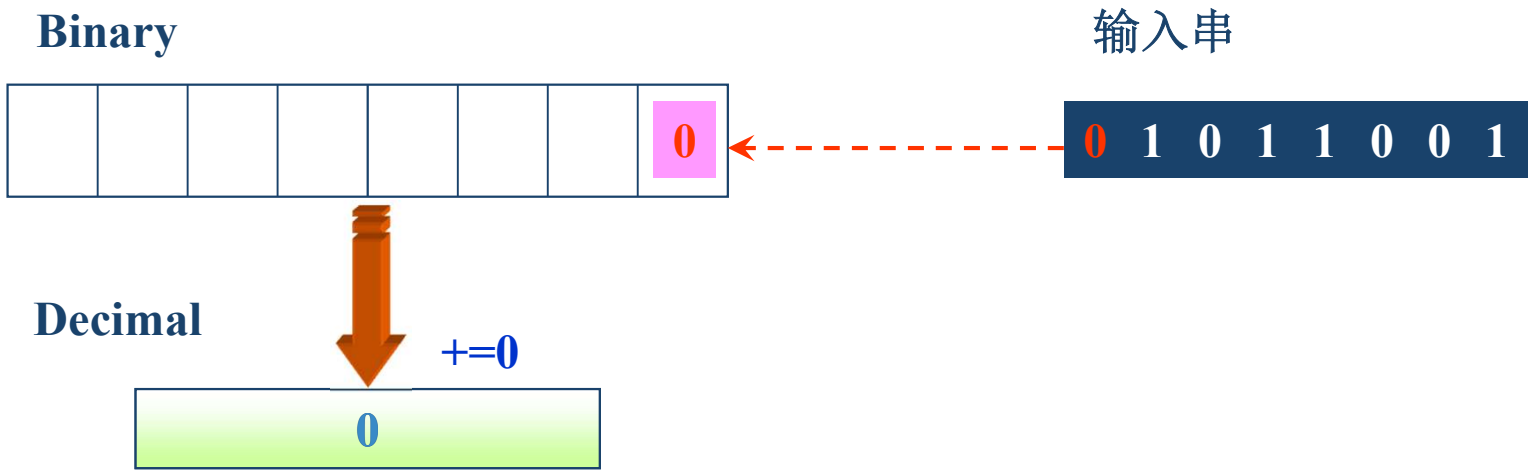
0

例 2-15 输入二进制数字串，转换成十进制正整数。

二进制数$b_n b_{n-1} \ldots b_2 b_1 b_0$有转换为十进制Dec的公式：

$$Dec = b_n \times 2^n + b_{n-1} \times 2^{n-1} + \ldots + b_2 \times 2^2 + b_1 \times 2^1 + b_0 \times 2^0$$

输入 — 移位 转换



例 2-15 输入二进制数字串，转换成十进制正整数。

二进制数$b_n b_{n-1} \ldots b_2 b_1 b_0$有转换为十进制Dec的公式：

$$Dec = b_n \times 2^n + b_{n-1} \times 2^{n-1} + \ldots + b_2 \times 2^2 + b_1 \times 2^1 + b_0 \times 2^0$$

输入 — 移位 转换

Binary **×2**

						0	0

输入串

1	0	1	1	0	0	1

Decimal ***=2**

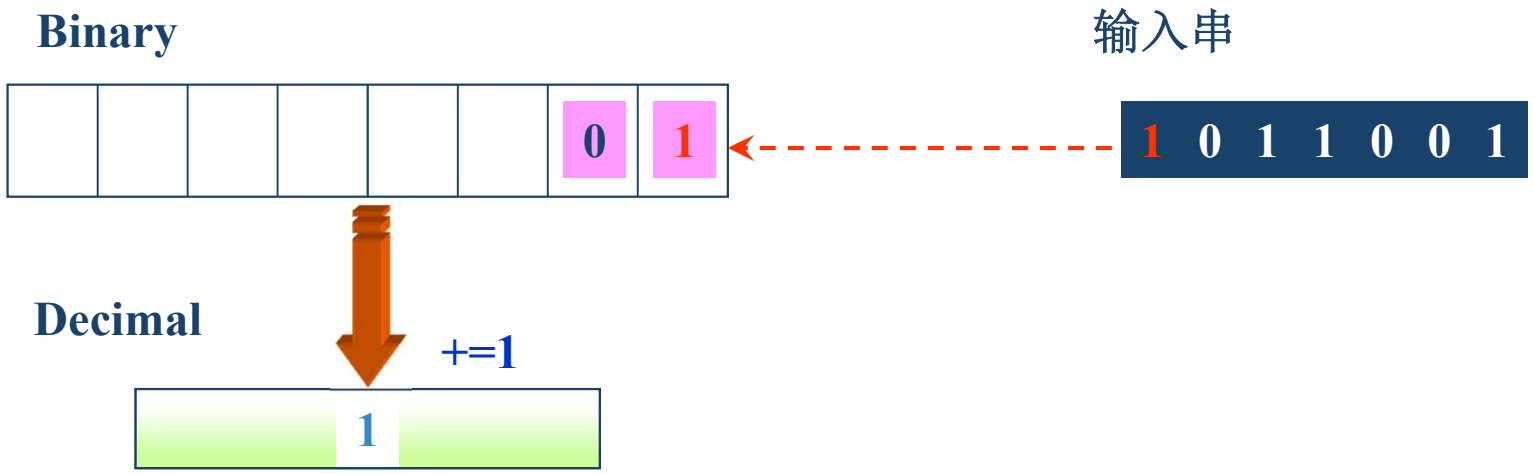
0

例 2-15 输入二进制数字串，转换成十进制正整数。

二进制数$b_n b_{n-1} \ldots b_2 b_1 b_0$有转换为十进制Dec的公式：

$$Dec = b_n \times 2^n + b_{n-1} \times 2^{n-1} + \ldots + b_2 \times 2^2 + b_1 \times 2^1 + b_0 \times 2^0$$

输入 — 移位 转换



例 2-15 输入二进制数字串，转换成十进制正整数。

二进制数$b_n b_{n-1} \ldots b_2 b_1 b_0$有转换为十进制Dec的公式：

$$Dec = b_n \times 2^n + b_{n-1} \times 2^{n-1} + \ldots + b_2 \times 2^2 + b_1 \times 2^1 + b_0 \times 2^0$$

输入 — 移位 转换

Binary **×2**

					0	1	0

输入串

0	1	1	0	0	1

Decimal ***=2**

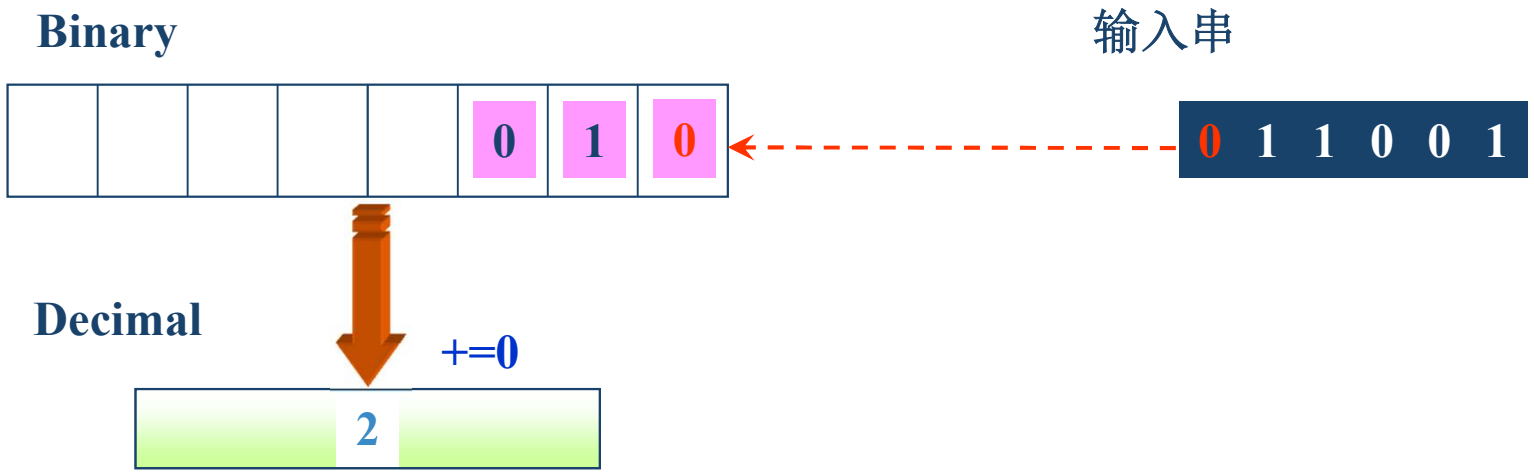
	2	

例 2-15 输入二进制数字串，转换成十进制正整数。

二进制数$b_n b_{n-1} \ldots b_2 b_1 b_0$有转换为十进制Dec的公式：

$$Dec = b_n \times 2^n + b_{n-1} \times 2^{n-1} + \ldots + b_2 \times 2^2 + b_1 \times 2^1 + b_0 \times 2^0$$

输入 — 移位 转换



例 2-15 输入二进制数字串，转换成十进制正整数。

二进制数$b_n b_{n-1} \ldots b_2 b_1 b_0$有转换为十进制Dec的公式：

$$Dec = b_n \times 2^n + b_{n-1} \times 2^{n-1} + \ldots + b_2 \times 2^2 + b_1 \times 2^1 + b_0 \times 2^0$$

输入 — 移位 转换

Binary **×2**

				0	1	0	*0*

输入串

1 1 0 0 1

Decimal ***=2**

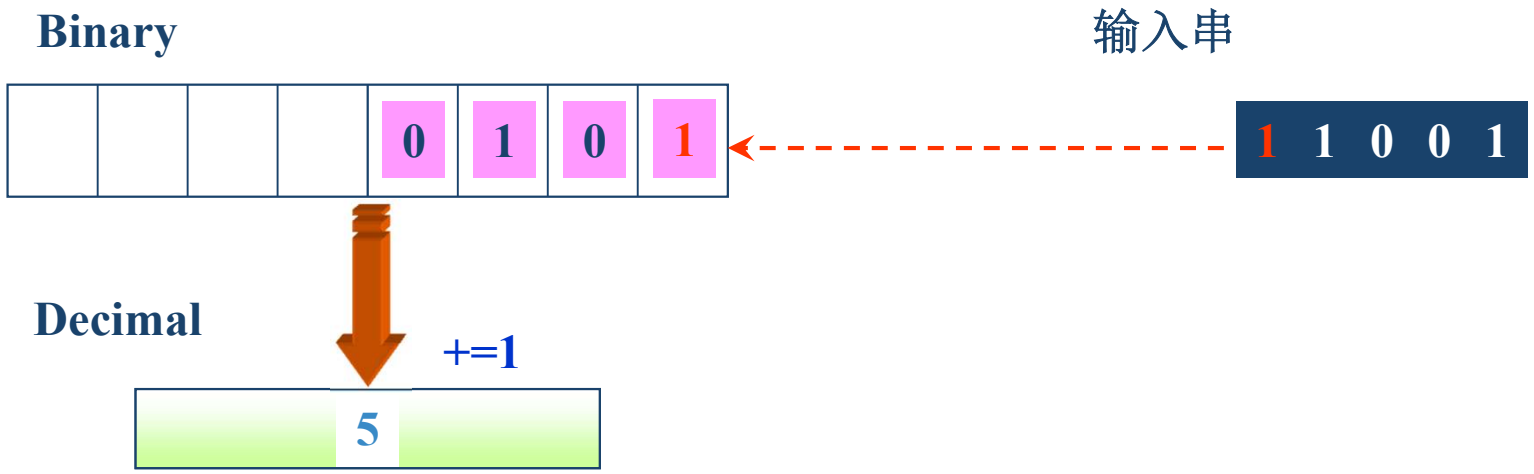
				4			

例 2-15 输入二进制数字串，转换成十进制正整数。

二进制数$b_n b_{n-1} \ldots b_2 b_1 b_0$有转换为十进制Dec的公式：

$$Dec = b_n \times 2^n + b_{n-1} \times 2^{n-1} + \ldots + b_2 \times 2^2 + b_1 \times 2^1 + b_0 \times 2^0$$

输入 — 移位 转换

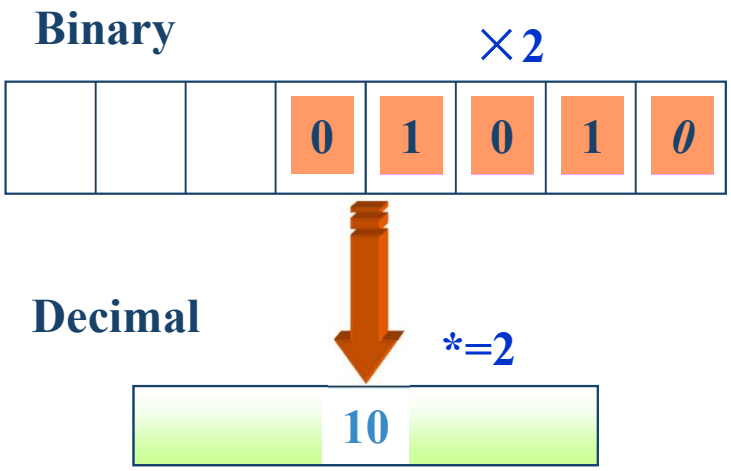


例 2-15 输入二进制数字串，转换成十进制正整数。

二进制数$b_n b_{n-1} \ldots b_2 b_1 b_0$有转换为十进制Dec的公式：

$$Dec = b_n \times 2^n + b_{n-1} \times 2^{n-1} + \ldots + b_2 \times 2^2 + b_1 \times 2^1 + b_0 \times 2^0$$

输入 — 移位 转换



输入串

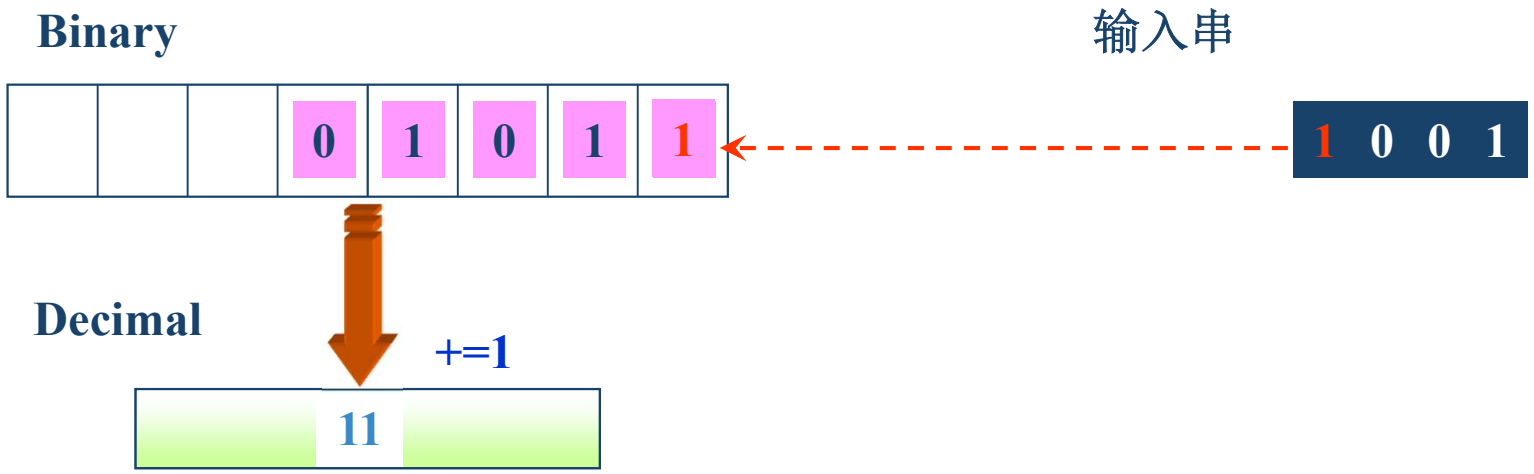
1 0 0 1

例 2-15 输入二进制数字串，转换成十进制正整数。

二进制数$b_n b_{n-1} \ldots b_2 b_1 b_0$有转换为十进制Dec的公式：

$$Dec = b_n \times 2^n + b_{n-1} \times 2^{n-1} + \ldots + b_2 \times 2^2 + b_1 \times 2^1 + b_0 \times 2^0$$

输入 — 移位 转换

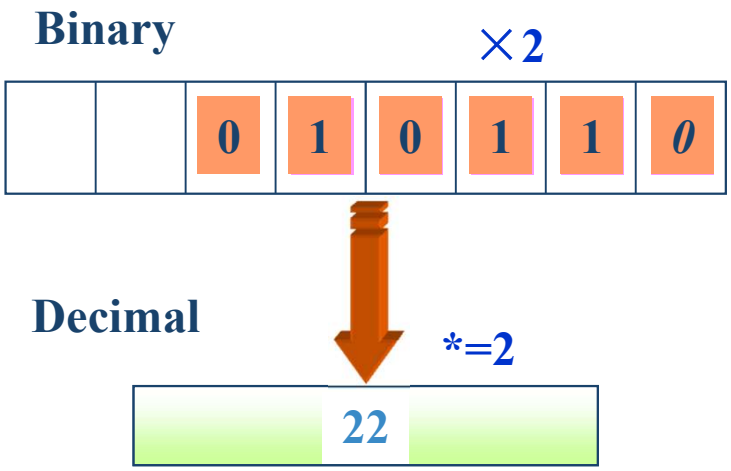


例 2-15 输入二进制数字串，转换成十进制正整数。

二进制数$b_n b_{n-1} \ldots b_2 b_1 b_0$有转换为十进制Dec的公式：

$$Dec = b_n \times 2^n + b_{n-1} \times 2^{n-1} + \ldots + b_2 \times 2^2 + b_1 \times 2^1 + b_0 \times 2^0$$

输入 — 移位 转换



输入串

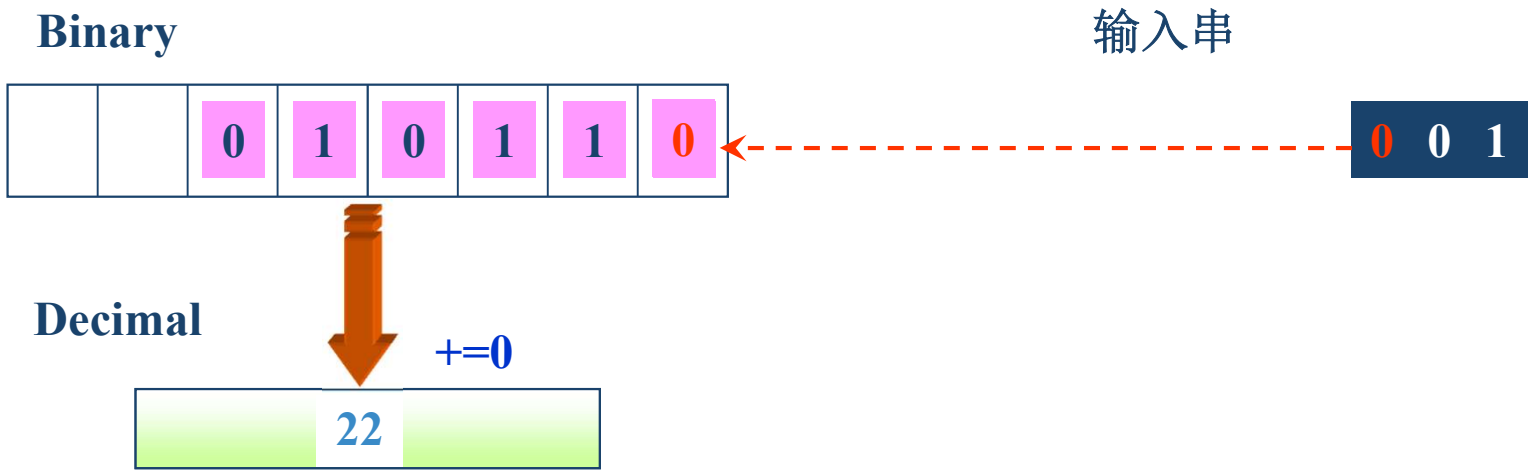
0 0 1

例 2-15 输入二进制数字串，转换成十进制正整数。

二进制数$b_n b_{n-1} \ldots b_2 b_1 b_0$有转换为十进制Dec的公式：

$$Dec = b_n \times 2^n + b_{n-1} \times 2^{n-1} + \ldots + b_2 \times 2^2 + b_1 \times 2^1 + b_0 \times 2^0$$

输入 — 移位 转换

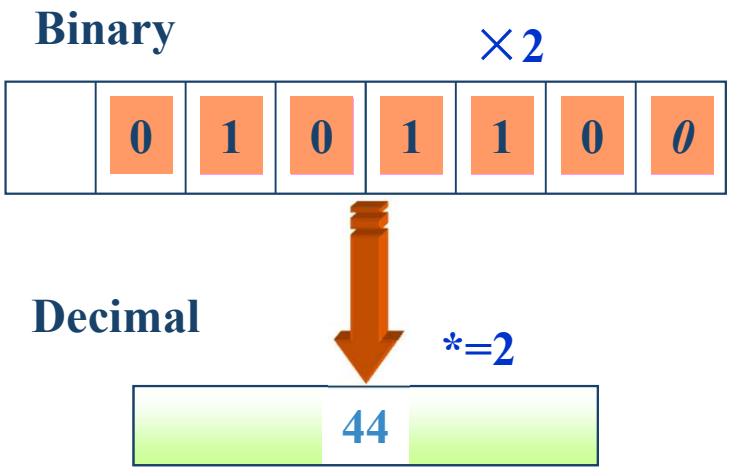


例 2-15 输入二进制数字串，转换成十进制正整数。

二进制数$b_n b_{n-1} \ldots b_2 b_1 b_0$有转换为十进制Dec的公式：

$$Dec = b_n \times 2^n + b_{n-1} \times 2^{n-1} + \ldots + b_2 \times 2^2 + b_1 \times 2^1 + b_0 \times 2^0$$

输入 — 移位 转换



输入串

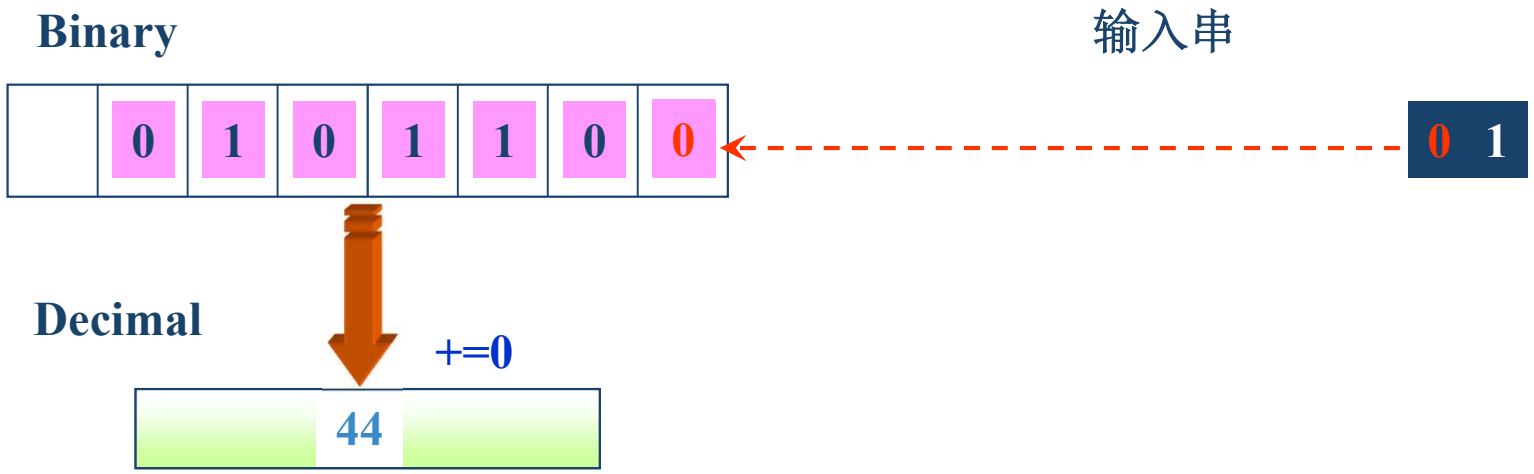
0 1

例 2-15 输入二进制数字串，转换成十进制正整数。

二进制数$b_n b_{n-1} \ldots b_2 b_1 b_0$有转换为十进制Dec的公式：

$$Dec = b_n \times 2^n + b_{n-1} \times 2^{n-1} + \ldots + b_2 \times 2^2 + b_1 \times 2^1 + b_0 \times 2^0$$

输入 — 移位 转换

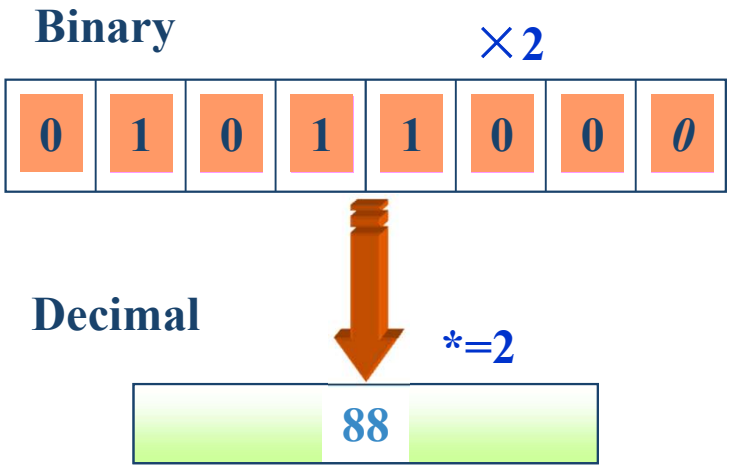


例 2-15 输入二进制数字串，转换成十进制正整数。

二进制数$b_n b_{n-1} \ldots b_2 b_1 b_0$有转换为十进制Dec的公式：

$$Dec = b_n \times 2^n + b_{n-1} \times 2^{n-1} + \ldots + b_2 \times 2^2 + b_1 \times 2^1 + b_0 \times 2^0$$

输入 — 移位 转换



输入串

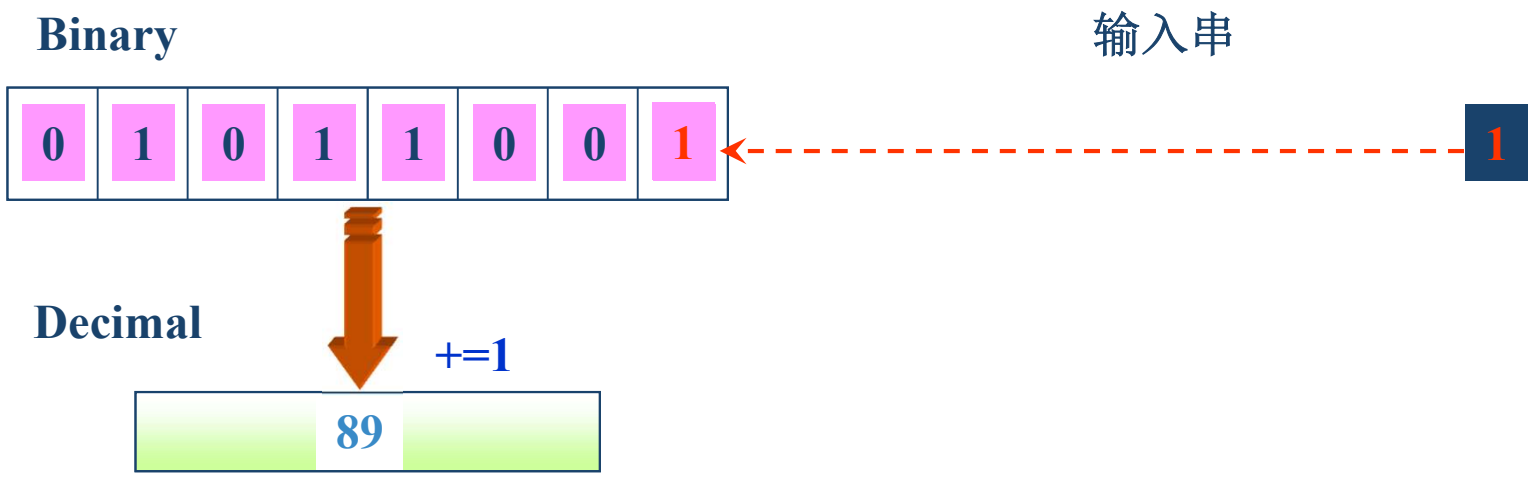
1

例 2-15 输入二进制数字串，转换成十进制正整数。

二进制数$b_n b_{n-1} \ldots b_2 b_1 b_0$有转换为十进制Dec的公式：

$$Dec = b_n \times 2^n + b_{n-1} \times 2^{n-1} + \ldots + b_2 \times 2^2 + b_1 \times 2^1 + b_0 \times 2^0$$

输入 — 移位 转换



例 2-15 输入二进制数字串，转换成十进制正整数。

二进制数$b_n b_{n-1} \ldots b_2 b_1 b_0$有转换为十进制Dec的公式：

$$Dec = b_n \times 2^n + b_{n-1} \times 2^{n-1} + \ldots + b_2 \times 2^2 + b_1 \times 2^1 + b_0 \times 2^0$$

输入 — 移位 转换

Binary

0	1	0	1	1	0	0	1

输入串

0 1 0 1 1 0 0 1

Decimal

89

// 例 2-15 输入二进制数字串，转换成十进制正整数

```
#include <iostream>
using namespace std;
int main()
{ int Dec = 0 ;
  char ch;
  cout << "Binary = " ;
  do
     { cin.get(ch) ;
     } while( ch !='0' && ch != '1' ) ;
  do
    { Dec += ch - '0';
      cin.get(ch);
      if ( ch=='0'||ch=='1' )
         Dec *= 2 ;
    } while( ch=='0'||ch== '1' );
  cout << "Decimal = " << Dec << '\n';
}
```

```
// 例 2-15  输入二进制数字串，转换成十进制正整数
#include <iostream>
using namespace std;
int main()
{ int Dec = 0 ;
  char ch;
  cout << "Binary = " ;
  do                    //略去前导符号，直至ch存放第一个合法数字
     { cin.get(ch) ;
     } while( ch !='0' && ch != '1' ) ;
  do
    { Dec += ch - '0';
      cin.get(ch);
      if ( ch=='0'||ch=='1' )
         Dec *= 2 ;
    } while( ch=='0'||ch== '1' );
  cout << "Decimal = " << Dec << '\n';
}
```

// 例 2-15 输入二进制数字串，转换成十进制正整数

```
#include <iostream>
using namespace std;
int main()
{ int Dec = 0 ;
  char ch;
  cout << "Binary = " ;
  do                          //略去前导符号，直至ch存放第一个合法数字
     { cin.get(ch) ;
     } while( ch !='0' && ch != '1' ) ;
  do                                  //循环，逐位转换
    { Dec += ch - '0';
      cin.get(ch);
      if ( ch=='0'||ch=='1' )
         Dec *= 2 ;
    } while( ch=='0'||ch== '1' );          //读入非0，非1字符时结束循环
  cout << "Decimal = " << Dec << '\n';
}
```

```
// 例 2-15  输入二进制数字串，转换成十进制正整数
#include <iostream>
using namespace std;
int main()
{ int Dec = 0 ;
  char ch;
  cout << "Binary = " ;
  do                    //略去前导符号，直至ch存放第一个合法数字
     { cin.get(ch) ;
     } while( ch !='0' && ch != '1' ) ;
  do                                //循环，逐位转换
   { Dec += ch - '0';               //把字符转换为数字，累加
     cin.get(ch);
     if ( ch=='0'||ch=='1' )
        Dec *= 2 ;
   } while( ch=='0'||ch== '1' );              //读入非0，非1字符时结束循环
  cout << "Decimal = " << Dec << '\n';
}
```

```
// 例 2-15  输入二进制数字串，转换成十进制正整数
#include <iostream>
using namespace std;
int main()
{ int Dec = 0 ;
  char ch;
  cout << "Binary = " ;
  do                        //略去前导符号，直至ch存放第一个合法数字
     { cin.get(ch) ;
     } while( ch !='0' && ch != '1' ) ;
  do                                  //循环，逐位转换
    { Dec += ch - '0';                //把字符转换为数字，累加
      cin.get(ch);                    //读入一位
      if ( ch=='0'||ch=='1' )
         Dec *= 2 ;
    } while( ch=='0'||ch== '1' );              //读入非0，非1字符时结束循环
  cout << "Decimal = " << Dec << '\n';
}
```

```
// 例 2-15  输入二进制数字串，转换成十进制正整数
#include <iostream>
using namespace std;
int main()
{ int Dec = 0 ;
  char ch;
  cout << "Binary = " ;
  do                        //略去前导符号，直至ch存放第一个合法数字
     { cin.get(ch) ;
     } while( ch !='0' && ch != '1' ) ;
  do                                //循环，逐位转换
    { Dec += ch - '0';              //把字符转换为数字，累加
      cin.get(ch);                  //读入一位
      if ( ch=='0'||ch=='1' )       //如果是0或1
         Dec *= 2 ;                 //已经转换的数据左移一位
    } while( ch=='0'||ch== '1' );           //读入非0，非1字符时结束循环
  cout << "Decimal = " << Dec << '\n';
}
```

```
// 例 2-15  输入二进制数字串，转换成十进制正整数
#include <iostream>
using namespace std;
int main()
{ int Dec = 0 ;
  char ch;
  cout << "Binary = " ;
  do                          //略去前导符号，直至ch存放第一个合法数字
     { cin.get(ch) ;
     } while( ch !='0' && ch != '1' ) ;
  do                                  //循环，逐位转换
    { Dec += ch - '0';                //把字符转换为数字，累加
      cin.get(ch);                    //读入一位
      if ( ch=='0'||ch=='1' )         //如果是0或1
         Dec *= 2 ;                   //已经转换的数据左移一位
    } while( ch=='0'||ch== '1' );              //读入非0，非1字符时结束循环
  cout << "Decimal = " << Dec << '\n';         //输出转换结果
}
```

```
// 例 2-15  输入二进制数字串，转换成十进制正整数
#include <iostream>
using namespace std;
int main()
{ int Dec = 0 ;
  char ch;
  cout << "Binary = " ;
  do                    //略去前导符号，直至ch存放第一个合法数字
     { cin.get(ch) ;
     } while( ch !='0' && ch != '1' ) ;
  do                                //循环，逐位转换
    { Dec += ch - '0';              //把字符转换
      cin.get(ch);                  //读入一位
      if ( ch=='0'||ch=='1' )       //如果是0或1
         Dec *= 2 ;                 //已经转换的
    } while( ch=='0'||ch== '1' );           //读
  cout << "Decimal = " << Dec << '\n';          //输出转换结果
}
```

2.3.3 for语句

语句形式

for (*表达式1; 表达式2 ; 表达式3*)

循环体 ;

2.3.3 for语句

语句形式

for (*表达式1; 表达式2 ; 表达式3*)

循环体 ;

关键字

2.3.3 for语句

语句形式

for (***表达式1***; *表达式2* ; *表达式3*)

循环体;

初始表达式

2.3.3 for语句

语句形式

for (*表达式1*; ***表达式2*** ; *表达式3*)

循环体;

循环控制
逻辑表达式

2.3.3 for语句

语句形式

for (*表达式1*; *表达式2* ; ***表达式3***)

循环体;

循环后置表达式

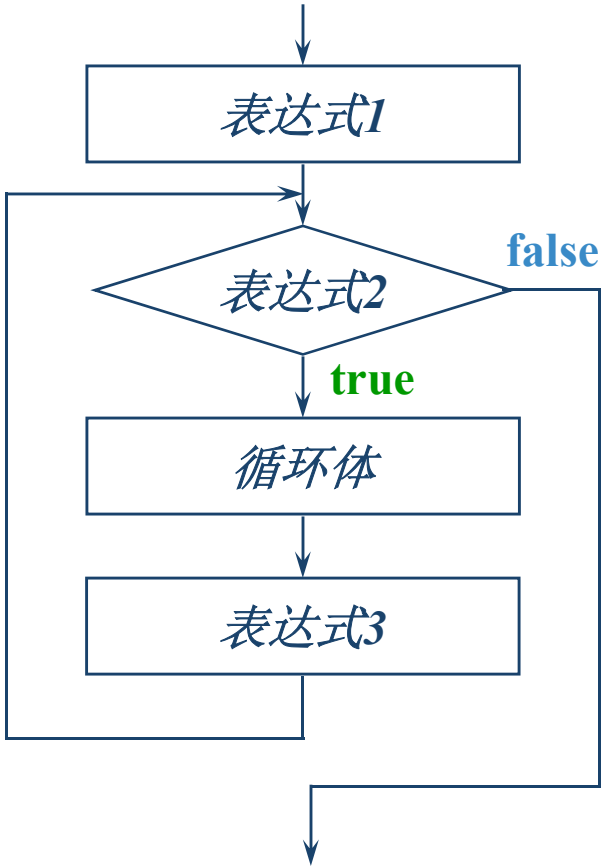
2.3.3 for语句

语句形式

for (*表达式1*; *表达式2* ; *表达式3*)

***循环体*;**

执行流程



2.3.3 for语句

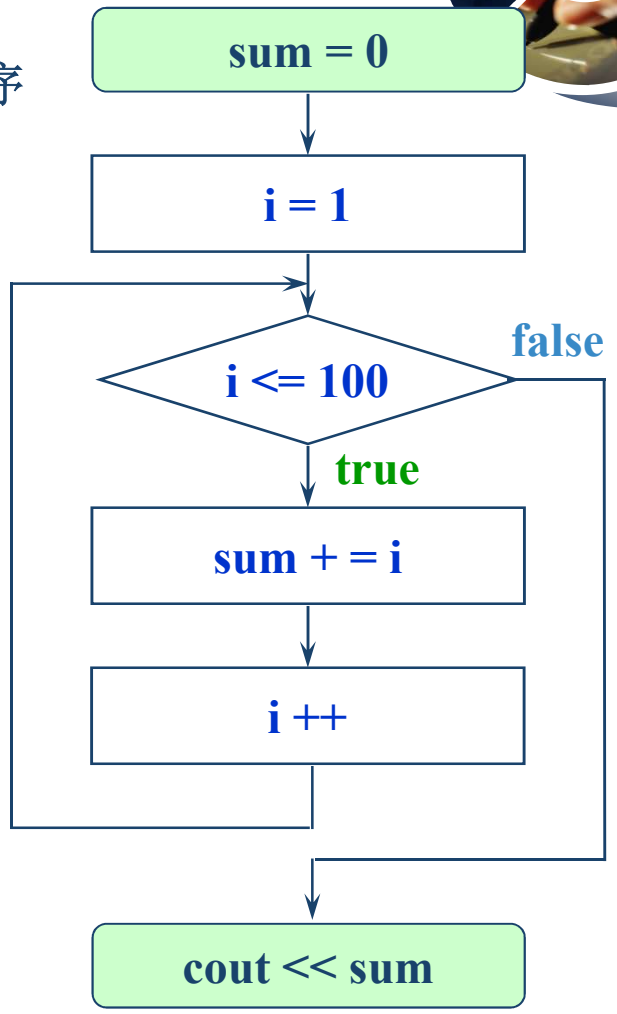
例如，用 for 语句的求和式 $sum = \sum_{i=1}^{100} i$ 的程序

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

2.3.3 for语句

例如，用 for 语句的求和式 $sum = \sum_{i=1}^{100} i$ 的程序

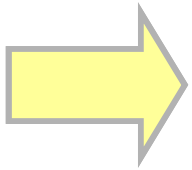
```
# include <iostream>
using namespace std ;
int main ()
{  int  i ,   sum = 0 ;
   for ( i =1 ;  i <= 100 ;  i ++ )
        sum + = i ;
   cout << " sum = " << sum << endl ;
}
```



2.3.3 for语句

$$sum = \sum_{i=1}^{n} i$$

```
for ( i =1 ; i < = n ; i + + )
  { sum = sum + i ; }
```



```
i = 1;
while ( i < = n )
    sum + = i + + ;
```

```
i = 1;
do
    sum + = i + + ;
while ( i < = n )
```

2.3.3 for语句

$$sum = \sum_{i=1}^{n} i$$

不同形式的for 语句结构

(1)
```
i =1 ;   //缺省表达式1
for ( ;  i <= n ;  i ++  )
  {  sum = sum  + i ; }
```

(2)
```
for ( i =1;  ;  i ++ )
   {  sum = sum  + i ;
      if  (i > n)  break ; }
   //缺省表达式2
```

(3)
```
for ( i =1;  i <= n ; )
   {  sum = sum  + i ;
      i + + ;  }
   //缺省表达式3
```

(4)
```
for
   ( i =1;  i <= n ;  sum + = i + + ) ;
   //缺省循环体
```

2.3.3 for语句

$$sum = \sum_{i=1}^{n} i$$

不同形式的 **for** 语句结构

(5) **for**

```
( i =1; sum + = i + +, i <= n ; ) ;
    //缺省表达式3和循环体
```

注意
逗号表达式

(6)

```
i = 1;
for ( ; ; )
{ sum + = i + + ;
  if ( i > n ) break ;
}
//缺省全部for 的表达式
```

第2章 C++简单程序设计（二）

（清华郑莉老师第2章）循环结构 之 for语句（8分钟）：

https://www.xuetangx.com/learn/THU08091000247/THU08091000247/12424464/video/23270418

例2-8：输入一个整数，求出它的所有因子


```
#include <iostream>
using namespace std;
int main() {
  int n;
  cout << "Enter a positive integer: ";
  cin >> n;
  cout << "Number  " << n << "   Factors  ";
  for (int k = 1; k <= n; k++)
    if (n % k == 0)
      cout << k << "  ";
  cout << endl;
  return 0;
}
```

运行结果1：

```
Enter a positive integer: 36
Number  36  Factors  1  2  3  4  6  9  12  18  36
```

运行结果2：

```
Enter a positive integer: 7
Number  7   Factors  1  7
```


那就从1开始数呗

for（初始语句； 表达式1； 表达式2 ）语句

循环前先求解　　为true时执行循环体　　每次执行完循环体后求解

例 2-18 判定素数

判定整数m是否素数

整数 m 是素数的条件： 除 1 和 m 外，没有其它因数。

从定义出发，采用*穷举法*逐个数据测试：

***穷举法*是一种重复型算法。**

基本思想是：对问题的所有可能状态一一测试，

直到　找到解

或　将全部可能状态都测试过为止。

例 2-18 判定素数

判定整数m是否素数

整数 m 是素数的条件： 除 1 和 m 外，没有其它因数。

从定义出发，采用*穷举法*逐个数据测试：

以 i = 2 ~ m-1 作除数，只要有一个 i 是 m 的因数，则 m 不是素数。

```
for ( int i = 2;  i < m ; i ++ )
    if ( m % i == 0 )    break ;
if ( m == i )
   m 是素数 ;
else
   m 不是素数;
```

例 2-18 判定素数

```
#include <iostream>
#include <cmath>
using namespace std ;
int main()
{ int i ;
  long  m ;
  cout <<"Please input a number:\n";
  cin >> m ;
  for ( i =2;  i < m;  i ++ )
      if ( m% i == 0 )  break ;
  if ( m == i )
    cout << m << " is prime." << endl ;
  else
    cout << m << " is not prime." << endl ;
}
```

例 2-18 判定素数

```
#include <iostream>
#include <cmath>
using namespace std ;
int main()
{ int i ;
  long  m ;
  cout <<"Please input a number:\n";
  cin >> m ;
  for ( i =2;  i < m;  i ++ )
     if ( m% i == 0 )  break ;
  if ( m == i )
    cout << m << " is prime." << endl ;
  else
    cout << m << " is not prime." << endl ;
}
```

找 m 的因数

例 2-18 判定素数

```
#include <iostream>
#include <cmath>
using namespace std ;
int main()
{ int i ;
  long  m ;
  cout <<"Please input a number:\n";
  cin >> m ;
  for ( i =2;  i < m;  i ++ )
      if ( m% i == 0 )  break ;
  if ( m == i )
    cout << m << " is prime." << endl ;
  else
    cout << m << " is not prime." << endl ;
}
```

判断 **m** 是否

被小于 **m** 的数整除

例 2-18 判定素数

```
#include <iostream>
#include <cmath>
using namespace std ;
int main()
{ int i ;
  long  m ;
  cout <<"Please input a number:\n";
  cin >> m ;
  for ( i =2;  i < m;  i ++ )
     if ( m% i == 0 )  break ;
  if ( m == i )
    cout << m << " is prime." << en
  else
    cout << m << " is not prime." <
}
```

例 2-18 判定素数

判定整数m是否素数

整数 m 是素数的条件：　　除 1 和 m 外，没有其它因数。

算法优化：

若 m 不是素数，有：m = i *j，i <= j，i <=$\sqrt{m}$ ，j >=$\sqrt{m}$

```
for ( int i = 2;  i <= sqrt( m ) ;  i ++ )
    if ( m %  i == 0 )    break ;
if (sqrt( m ) < i )
   m 是素数 ;
else
   m 不是素数;
```

例 2-18 判定素数

```
#include <iostream>
#include <cmath>
using namespace std ;
int main()
{ int i ;
  long  m;
  cout <<"Please input a number:\n";
  cin >> m ;
  double sqrtm = sqrt( m ) ;              //函数sqrt是double类型
  for ( i =2;  i <= sqrtm ;  i ++ )
     if ( m% i == 0 )  break ;
  if ( sqrtm < i )
    cout <<m<<" is prime."<<endl ;
  else
    cout << m << " is not prime." <<endl ;
}
```

例 2-18 判定素数

```
#include <iostream>
#include <cmath>
using namespace std ;
int main()
{ int i ;
  long  m;
  cout <<"Please input a number:\n";
  cin >> m ;
  double sqrtm = sqrt( m ) ;                    //函数sqrt是double类型
  for ( i =2;  i <= sqrtm ;  i ++ )
     if ( m% i == 0 )  break ;
  if ( sqrtm < i )
    cout <<m<<" is prime."<<endl ;
  else
    cout << m << " is not prime." <<endl ;
}
```

求 m 的平方根

例 2-18 判定素数

```
#include <iostream>
#include <cmath>
using namespace std ;
int main()
{ int i ;
  long  m;
  cout <<"Please input a number:\n";
  cin >> m ;
  double sqrtm = sqrt( m ) ;                    //函数sqrt是double类型
  for ( i =2; i <= sqrtm ;  i ++ )
     if ( m% i == 0 )  break ;
  if ( sqrtm < i )
    cout <<m<<" is prime."<<endl ;
  else
    cout << m << " is not prime." <<endl ;
}
```

注意循环条件

例 2-18 判定素数

```
#include <iostream>
#include <cmath>
using namespace std ;
int main()
{ int i ;
  long  m;
  cout <<"Please input a number:\n";
  cin >> m ;
  double sqrtm = sqrt( m ) ;                    //函数sqrt是double类型
  for ( i =2;  i <= sqrtm ;  i ++ )
     if ( m% i == 0 )  break ;
  if ( sqrtm < i )
    cout <<m<<" is prime."<<endl ;
  else
    cout << m << " is not prime." <<endl ;
}
```

判断素数条件

例 2-19 找出100~200之间的所有素数

分析：

用循环 **for m : 100 ~ 200,** 判断每个 **m** 是否素数。

```
for ( m = 101;  m <= 200 ;  m+=2  )
    if ( m是素数 )
        输出 m ;
```

例 2-19 找出100~200之间的所有素数

除2外
所有偶数不是素数

分析：

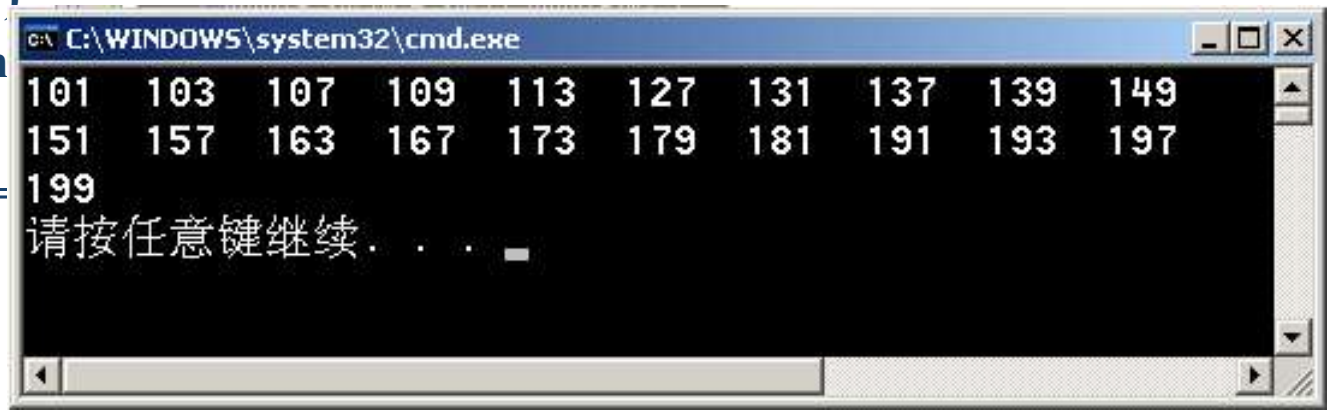
用循环 for m : 100 ~ 200, 判断每个 m 是否素数。

```
for ( m = 101;  m <= 200;  m+=2 )
    if ( m是素数 )
        输出 m ;
```

例 2-19 找出100—200之间的所有素数

```
#include< iostream >
#include< cmath >
using namespace std ;
int main()
{ long m ;  int i, k = 0 ;
  for( m = 101; m <= 200 ; m += 2 )
    { double sqrtm = sqrt( double(m) ) ;
      for( i = 2 ; i <= sqrtm ; i ++ )
        if( m % i == 0 )  break  ;
      if( sqrtm < i  )
        {  cout << m
           k ++ ;
           if( k%10 =
        }
    }
  cout << endl ;
}
```

参数类型转换



第2章 小测验题

单选题 1分

设置

（2.1）下面几个表达式中哪个的值等于或者最接近于1.0。

A 9.0/5

B 9/5

C float(9)/5

D 9/(double)5

提交

单选题 1分

设置

（2.1）设x为整型变量，不能正确表达数学关系1<x<5 的C++逻辑表达式是（ ）。

A 1< x <5

B x==2||x==3||x==4

C 1<x && x<5

D !(x<=1) && !(x>=5)

提交

单选题 1分

设置

（2.1）已知int i=1, j=2; 则表达式i++ +j的值为（ ）。

A 1

B 2

C 3

D 4

提交

GU CN Computer Networks Institute

单选题 1分

设置

（2.1）执行下列语句后，x和y的值是（ ）。

```
int x, y;
x = y = 1; ++x || ++y;
```

A 1和1

B 1和2

C 2和1

D 2和2

提交

GU CN Computer Networks Institute

单选题 1分

设置

（2.1）已知int x=5; 执行下列语句后，x的值为（ ）。

x += x -= x * x;

A 20

B 25

C -40

D 40

提交

单选题 1分

设置

（2.1）有语句int a=1, b=2; 以下正确的输出语句是（ ）。

A cout<<a=a+b<<endl;

B cout<<a>b?a:b<<endl;

C cout<<(hex)a+b;

D cout<<&a<<endl<<a<<endl;

提交

单选题 1分

设置

（2.1）设int a=1, b=2, c=3, d=4; 则以下条件表达式的值为（ ）。

a < b ? a : c < d ? c : d

A 1

B 2

C 3

D 4

提交

GU CN Computer Networks Institute

单选题 1分

设置

（2.2）已知 int i=0, x=1, y=0; 在下列选项中，使i的值变成1的语句是(　　)。

A　if(x&&y) i++;

B　if(x==y) i++;

C　if(!x) i++;

D　if(x||y) i++;

提交

单选题 1分

设置

（2.2）已知 int i=0, x=1, y=0; 在下列选项中，使i的值变成1的语句是(　　)。

A　if(x) {if(y) i=1; else i=0; }

B　if(x) {if(y) i=1; } else i=0;

C　if(x) i=1; else { if(y) i=0; }

D　if(x) i=0; else { if(y) i=1; }

提交

单选题 1分

设置

（2.2）执行下列语句后，输出显示为（ ）。

```
char ch='A';
switch( ch )
{  case 'A' : ch++;
   case 'B' : ch++;
   case 'C' : ch++;
}
cout<<ch<<endl;
```

A A

B B

C C

D D

提交

GU CN Computer Networks Institute

单选题 1分

设置

（2.2）设i=2，执行下列语句后i的值为（ ）。

```
switch( i )
{  case 1 : i ++;
   case 2 : i --;
   case 3 : ++ i; break;
   case 4 : -- i;
   default : i ++;
}
```

A 1

B 2

C 3

D 4

提交

GU CN Computer Networks Institute

单选题 1分

设置

（2.3）已知 int i=0，x=0; 在下面while语句执行时循环次数为（ ）。

while(!x && i< 3) { x++; i++; }

A 1

B 2

C 3

D 4

提交

单选题 1分

设置

（2.3）已知 int i=3; 在下面do-while语句执行时的循环次数为（ ）。

```
do { i--; cout<<i<<endl; }while( i!=1);
```

A 1

B 2

C 3

D 4

提交

GU CN Computer Networks Institute

单选题 1分

设置

(2.3) 下面for语句执行时的循环次数为（ ）。

```
int i, j;
for ( i=0, j=5; i=j; )
{  cout<<i<<j<< endl; i++; j--; }
```

A 0

B 5

C 10

D 无限

提交

单选题 1分

设置

(2.3) 以下程序段形成死循环的是（ ）。

A `int x; for( x=0; x<3; ) { x++; };`

B `int i=3; for(; i; i-- );`

C `int k = 0; do { ++k; } while( k>=0 );`

D `int a=5; while( a ) { a--; };`

提交

单选题 1分

设置

(2.4) 以下程序段输出结果是（ ）。

```
int i, n=0;
for(i=0; i<10; i++)
{  if( i%3 ) break;
   n++;
}
cout<<n<<endl;
```

A 1

B 2

C 3

D 4

提交

GU CN Computer Networks Institute

单选题 1分

设置

(2.4) 以下程序段输出结果是（ ）。

```
int i, n=0;
for(i=0; i<10; i++)
{  if( i%3 ) continue;
   n++;
}
cout<<n<<endl;
```

A 1

B 2

C 3

D 4

提交

GU CN Computer Networks Institute

单选题 1分

设置

(2.4) 以下程序段输出结果是（ ）。

```
int i, n=0;
for(i=0; i<10; i++)
{  if( i>2 ) goto out;
   n++;
}
out: cout<<n<<endl;
```

A 1

B 2

C 3

D 4

提交

第2章 习题选讲

《习题与解答》部分程序练习题