课后学习视频

慕课视频片段

视频名称：格式化输入输出函数a

温馨提示：此视频框在点击“上传手机课件”时会进行转换，用手机进行观看时则会变为可点击的视频。此视频框可被拖动移位和修改大小

遇到了问题？

环境装不好？
Visual C++ 2010 Express is not the only choice for you

总有bug？
先看错误提示，尝试自己解决，群内交流

小例子

```
[nnbao@enine-ahu-asipp unix_test]$ cat shuru.c
#include <stdio.h>

void main(void)
{
  int i;
  printf("Please input a number:\n");
  scanf("%d",&i);
  printf("You number is %d:\n",i);
}
```

注意用户输入的格式

小例子

```
[nnbao@enine-ahu-asipp unix_test]$ cat shuchu.c
#include <stdio.h>

void main(void)
{
  int i;
  printf("Please input a number:\n");
  scanf("%d",&i);
  printf("You number is %d:\n",i+1);
}
```

输出值 i+1

第二章

数据及运算

预热问题

- 什么是变量和常量？为什么要有变量？
- 目前你见过的C 的数据类型?
- 目前你见过的C定义变量的方式?
- 目前你了解的运算有哪些？

第二章 数据及运算

重点

1. 理解C语言的基本数据类型
2. 学会正确书写整型常量、实型常量、字符型常量、字符串常量
3. 学会正确定义及使用基本数据类型变量
4. 学会正确使用算术、关系、逻辑、条件、赋值、逗号等运算符
5. 理解C语言表达式的构成与计算
6. 理解数据类型的自动转换
7. 掌握基本数据类型数据的输入/输出

难点

1. 学会正确书写C语言表达式解决常见的数学问题
2. 正确理解含有多个运算的C语言表达式运算的优先级
3. 正确理解格式输入/输出函数的“格式字符串”

常量与变量数据

int first=68;。。。 常量

- 在程序执行期间，其值不可改变的量称为常量
- 常量无需说明就可直接书写、引用。

求最大数的例子中int max；max =a; max=b; max=c;。。。 变量

- 变量是用来存放程序运行过程中的数据，例如程序运行过程中的输入数据、计算获得的中间结果和最终结果。变量中存放的数据允许改变。
- 为了区别不同的变量，每个变量都要有自己的名称，称为变量名。

2.1 C语言的数据类型

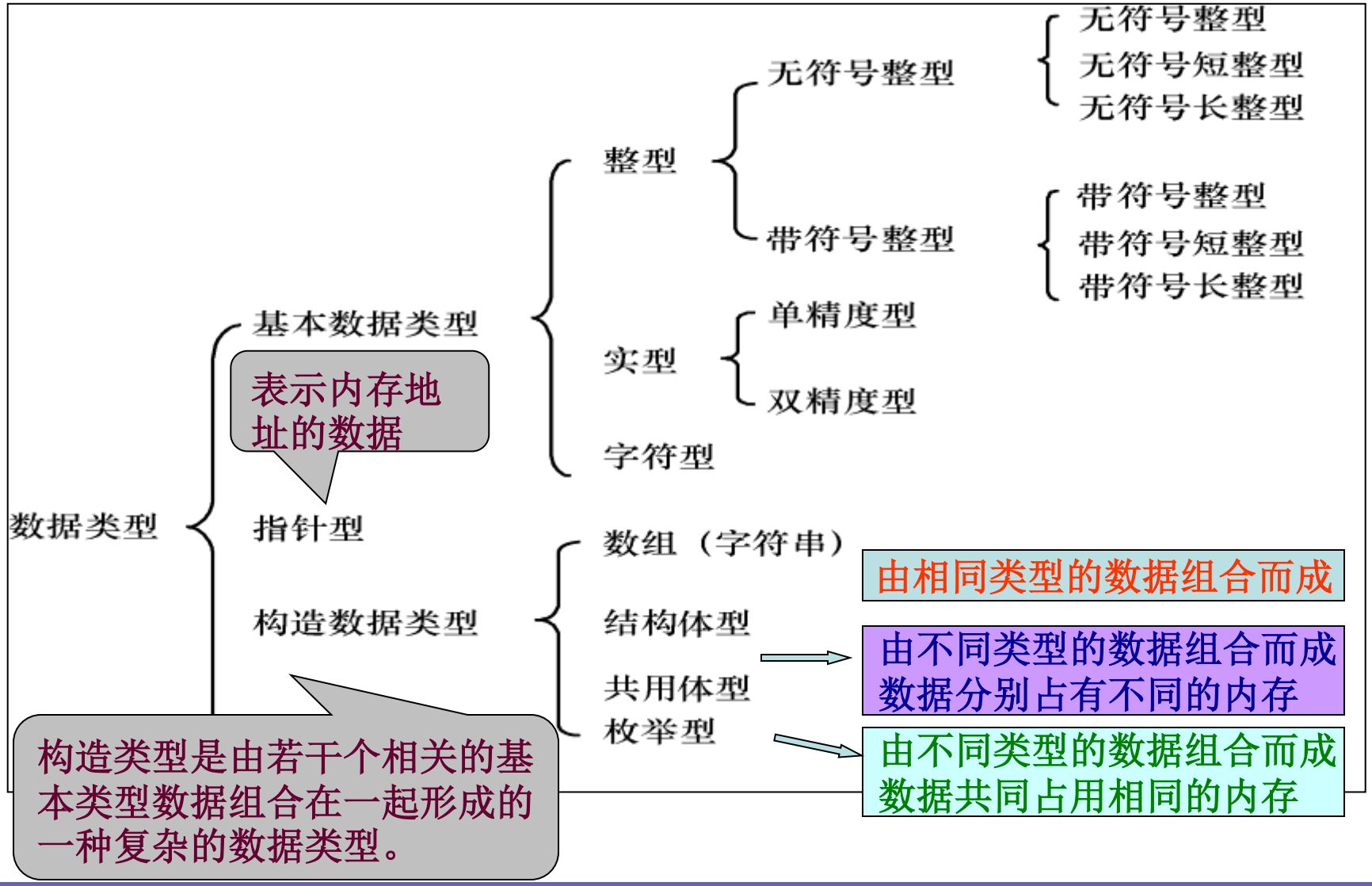
- **数据类型**

是指数据存储和加工时的特征

- **存储特征:**
  - 数据在内存中要占有多少字节
- **加工特征:**
  - 数据能进行哪种计算

Num	name	sex	age	score	addr
Y10010	Lining	M	18	87.5	Beijing

C语言的数据类型


数据类型
基本数据类型
整型
无符号整型
无符号整型
无符号短整型
无符号长整型
带符号整型
带符号整型
带符号短整型
带符号长整型
实型
单精度型
双精度型
字符型
表示内存地址的数据
指针型
构造数据类型
数组（字符串）
由相同类型的数据组合而成
结构体型
共用体型
由不同类型的数据组合而成
数据分别占有不同的内存
枚举型
由不同类型的数据组合而成
数据共同占用相同的内存
构造类型是由若干个相关的基本类型数据组合在一起形成的一种复杂的数据类型。

关于数据类型的说明

- 程序中用到的每个数据都要首先确定它们的数据类型。
- 任何一种数据类型的数据都要在内存中分配若干个字节，用于存放该数据。
- 数据占用的内存字节数称为该数据的“数据长度”。不同类型数据其长度可能不同。

数据类型关键字

- 数字？字母？
- C语言基本类型关键字

最初关键字（K&R）	新增关键字（C90标准）	新增关键字（C99标准）
int	signed	_Bool
long	void	_Complex
short		_Imaginary
unsigned		
char		
float		
double		

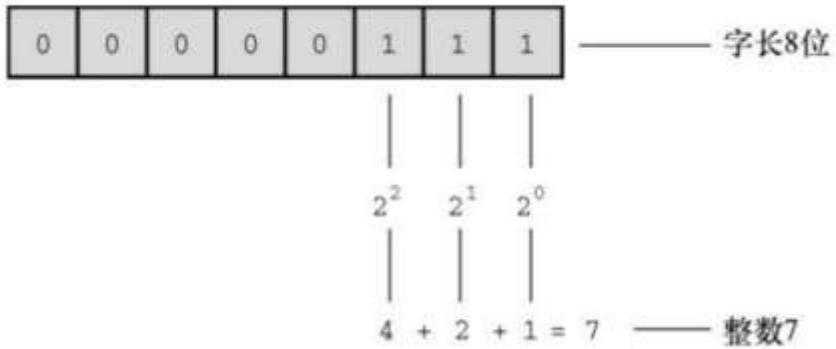
- 按计算机的储存方式可分为两大基本类型：整数类型和浮点数类型

整数

整数，没有小数部分的数，包括正整数、负整数和0 。

计算机以二进制数字储存整数，

例如，整数7以二进制写是111。



补充知识：位、字节和字

位、字节和字是描述计算机数据单元或存储单元的术语。

最小的存储单元是位（bit），可以储存0或1；字节（byte）是常用的计算机存储单位。对于几乎所有的机器，1字节均为8位。字（word）是设计计算机时给定的自然存储单位。

整数

整数可以写成不同数制的形式：

十六进制整数：以0x开头，由数字（0-9，a-f）表示

$0x123=(123)_{16}=(291)_{10}$ *不区分大小写！*

八进制整数：以数字0开头，由数字（0-7）表示

$0123=(123)_8=1\times8^2+2\times8^1+3\times8^0=(83)_{10}$

十进制整数：由数字（0-9）表示，如：123

短整型数用2字节存储，数据范围为：$-2^{15}\sim(2^{15}-1)$，（-32768~32767，无符号：0~65535）。普通整形、长整型常量在计算机中占用4个字节，能够表示的数值范围是：$-2^{31}\sim2^{31}-1$。(-2147483648~2147483647)

使用和显示不同进制的整数

```
//Present numbers in different format

#include <stdio.h>
int main(void)
{
  int x = 100;
  printf("here are dec, octal, hex\n");
  printf("%d,%o,%x\n",x,x,x);
  printf("%d,%#o,%#x\n",x,x,x);
  return 0;
}
~
```

```
[nnbao@enine-ahu-asipp unix_test]$ ./a.out
here are dec, octal, hex
100,144,64
100,0144,0x64
```

实型常量

实型常量是带小数点的实数，也称为“浮点数”。

实型常量只使用十进制数，它的书写方法有两种：

（1）一般形式的实数。

- 由整数部分、小数点、小数部分组成，正数前面的+号可以省略。

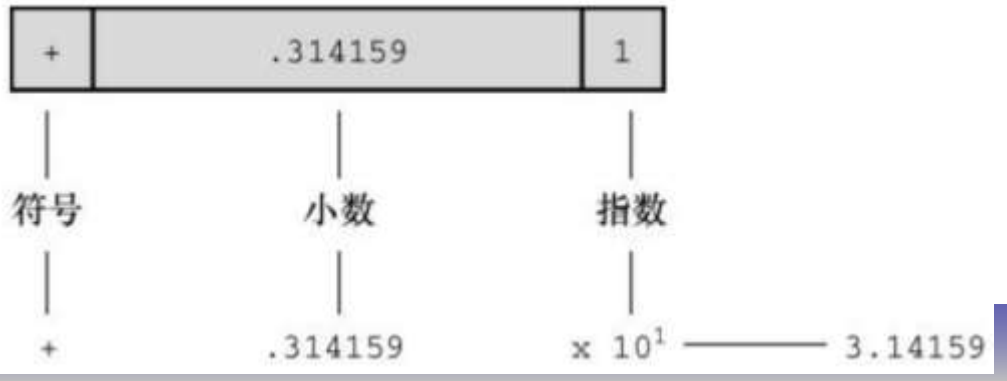
  -99.99 +16.0 -.066 660. 3.14159

（2）指数形式的实数。

- 由尾数部分、字母“E”（大小写均可）、指数部分组成。（科学计数法）

  +31.4e-1 .314e+1 -314.E-2 1E-5

例如，以浮点格式（十进制）储存π的值



实型常量

- C语言中的浮点类型有float、double和long double 类型
- 第1列是一般记数法；第2列是科学记数法；第3列是指数记数法（e记数法）

数字	科学记数法	指数记数法
1000000000	1.0×10^{9}	1.0e9
123000	1.23×10^{5}	1.23e5
322.56	3.2256×10^{2}	3.2256e2
0.000056	5.6×10^{-5}	5.6e-5

- C标准规定，float类型必须至少能表示6位有效数;
- double和float类型的最小取值范围相同，但至少必须能表示10位有效数字;
- long double类型至少与double类型的精度相同。

字符常量： ASCII表+扩充ASCII（汉字）

a.用一对单引号括起来的单个字符：'8'、'+'

b.值：对应ASCII码值 'A' ⇔ 65

c. 可以参加各种运算 'Z'＋ 'a' - 'A' ⇔ 'z'

- 可见字符（如字母、数字、标点符号等）
  - 使用单引号括住字符
- 不可见字符（如回车换行键、左退一格键等控制字符）。
  - C语言提供了另一种表示方法，即转义字符。

打印字母对应的ASCII值

```
[nnbao@enine-ahu-asipp unix_test]$ cat charcode.c
#include <stdio.h>

void main(void)
{
  char ch;
  printf("Please enter a character:\n");
  scanf("%c",&ch);
  printf("The code for %c is %d.\n",ch,ch);

}
```

```
[nnbao@enine-ahu-asipp unix_test]$ ./a.out
Please enter a character:
A
The code for A is 65.
```

转义字符：由“反斜杠字符”开始，后跟单个字符或若干个字符组成。（注意：不易理解，需认真体会）

转义字符表

转义字符	对应字符	转义字符	对应字符
\n	回车换行符	\a	响铃符号
\t	Tab符	\'	单引号
\0	空（字符串结束符）	\"	双引号
\b	左退一格符	\\	反斜杠
\r	回车符(回到行首)	\ddd	1～3位8进制数ddd对应的符号
\f	换页符	\xhh	1～2位16进制数hh对应的符号

转义字符说明

- “ \ ”在转义字符中作为特殊字符使用
- “ ' ” 作为字符常量的标记
- “ " ” 作为字符串常量的标记

➢ 用到这三个字符时，必须写成转义字符形式：

字符	\	'	"
表示	\\	\'	\"

“转义字符”也是单个字符：

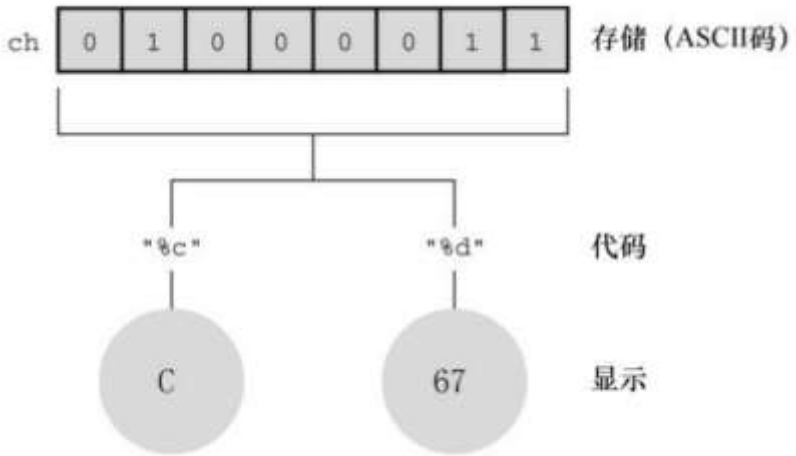
➢如：‘\n’、‘\t’、‘\a’、‘\123’、‘\0’、‘\x61’、‘\\’、‘\’‘、‘\”’都表示一个字符常量。

其他注意事项

- C语言规定，字母区分大小写，所以‘c’和‘C’是不同的字符常量。

- 字符常量在计算机中占用1个字节，存放的内容是该字符对应的ASCII代码值。

- 打印char类型变量的方式
- %d转换 整数
- %c转换 字符



其他注意事项

字符	‘0’	‘1’	‘A’	‘B’	‘a’	‘b’
ASCII值	48	49	65	66	97	98

- 字符常量也可以看成是“整型常量”参与运算，其值就是ASCII代码值。
  - ‘A’+5，‘C’-1，‘1’+2分别等于整数值70，66，51。

- 凡数值在0～127之间的正整数都可以看成是字符常量，例如70，66，51可以当作字符常量‘F’，‘B’，‘3’。

```c
//Program: EG0202.c
//Description: 演示字符常量的使用
#include <stdio.h>
void main(void)
{
    printf(" abc\tde\n");
    printf("ijk\tK\bM\n");
    printf("\'%c\'=%d\t\'%c\'=%d\n",'A','A','a','a');
}
```

```
C:\WINDOWS\system32\cmd.
 abc     de
ijk      M
'A'=65   'a'=97
请按任意键继续. . .
```

程序首先输出两个空格，然后输出字符 abc，之后跳过一个制表位（一般每个制表位有 8 列），在第 9 列输出字符 d、e；接着换行到下一行行首输出。下一行开始输出字符 i j k 后跳到下一个制表位输出字符 L，转义字符'\b'将光标回退一格，在字符 L 位置重新输出字符 M，接着换行到下一行行首，输出'A'=和字符 A 的 ASCII 值 65，在下一制表位输出'a'=和字符 a 的 ASCII 值 97。

字符串常量

- 字符串常量是用两个双引号（“）前后括住的若干个字符（包括转义字符）。字符串常量经常简称为“字符串”。

- “C”、“123”、“-12.3”、“\\ccw\\”、“\n\n”、“\41\x42”、“AB□”、“□”，“”都是字符串。其中的□表示空格字符。

字符串常量说明

- 注意空格字符也是字符串中的有效字：
  - " AB□ "表示字符串中有3个字符，
  - " □ "则表示字符串中只有一个空格字符。
  - ""中没有字符，称为空字符串。

- 字符串中的字母是区分大小写的，所以" C "和" c "是不同的字符串。

'C'和"C"含义是否一样？

注意字符常量和字符串的区别，'C'是字符常量，而"C"是字符串

字符串常量说明

- 字符串中所有字符的个数称为字符串长度，其中每个转义字符只能当做一个字符。

【例2-11】写出下列字符串长度。

"C"	"12.3"	"\\ccw\\"	"\n\n"	"\41\x42"	"AB□"	""
1	4	5	2	2	3	0

字符串常量说明

- 字符串由单个字符组成，每个字符串在内存中占用的字节数=字符串长度+1。
- 其中最后一个字节存放的字符称为“空字符”，其值为0，书写时常用转义字符“\0”来表示，称为“字符串结束标记”。

“AHU”

A	H	U	\0

“CHINA”

C	H	I	N	A	\0

符号常量

- 通俗地说，符号常量是给常量取一个名字。程序中凡是使用这个常量的地方，都可以写成对应常量的名字。
- 符号常量的定义格式如下：

  #define 符号常量 常量

- 其中的符号常量是任意选定的标识符，其中的字母习惯上约定用大写。
- 其中的常量可以是整型、实型、字符型，也可以是字符串。
- 这个定义通常是放在程序的开头，后不接分号。

EG0203.cpp

(全局范围)

```
// Program: EG0203.c
// Description: 演示符号常量的使用
#include <stdio.h>
#define PI 3.14159
void main(void)
{
    printf("半径：%f\n",1.0);
    printf("圆周长：%f\n",2*PI*1.0);
    printf("圆半径：%f\n",PI*1.0*1.0);
}
```

```
C:\WINDOWS\system32\cmd.
半径：1.000000
圆周长：6.283180
圆半径：3.141590
请按任意键继续. . .
```

说明： 程序中用#define命令行定义PI代表常量3.14159，此后凡在本文件中出现的PI都代表3.14159，可以和常量一样进行运算

在程序中使用符号常量有两点好处：

- 为阅读程序提供了方便，例如将3.14159定义成PI，就很容易理解常量是圆周率，该程序是求解一个圆的面积。

- 修改程序方便，当程序中多处使用了某个常量而又要修改该常量时，修改的操作十分繁琐，而且漏改了一处，程序运行结果就会出错。如果将这样的常量定义成符号常量，则只要修改定义就可以了。

表2-2 基本数据类型符及其含义

类型说明符	比特（字节）	数的范围
char	8（1）	$-2^7 \sim 2^7-1$ -128~127
unsigned char	8（1）	$0 \sim 2^8-1$ 0~255
short int	16（2）	$-2^{15} \sim 2^{15}-1$ -32768~32767
unsigned short int	16（2）	$0 \sim 2^{16}-1$ 0~65525
int/long	32（4）	$-2^{31} \sim 2^{31}-1$
unsigned int/long	32（4）	$0 \sim 2^{32}-1$（全0-全1）
float	32（4）	$-10^{38} \sim 10^{38}$ 有效数字7位
double	64（8）	$-10^{308} \sim 10^{308}$有效数字15位

EX02_sizeof.cpp

(全局范围) main(void)

```
#include <stdio.h>
int main(void)
{
    printf("Type int has a size of %d bytes.\n",sizeof(int));
    printf("Type char has a size of %d bytes.\n",sizeof(char));
    printf("Type long has a size of %d bytes.\n",sizeof(long));
    printf("Type long long has a size of %d bytes.\n",sizeof(long long));
    printf("Type float has a size of %d bytes.\n",sizeof(float));
    printf("Type double has a size of %d bytes.\n",sizeof(double));
    printf("Type long double has a size of %d bytes.\n",sizeof(long double));
}
```

```
Type int has a size of 4 bytes.
Type char has a size of 1 bytes.
Type long has a size of 4 bytes.
Type long long has a size of 8 bytes.
Type float has a size of 4 bytes.
Type double has a size of 8 bytes.
Type long double has a size of 8 bytes.
请按任意键继续. . .
```

2.4 运算符与表达式

2.4.1 C语言的运算符
2.4.2 运算符的优先级和结合性
2.4.1表达式计算中数据类型的自动转换

2.4.1 C语言的运算符

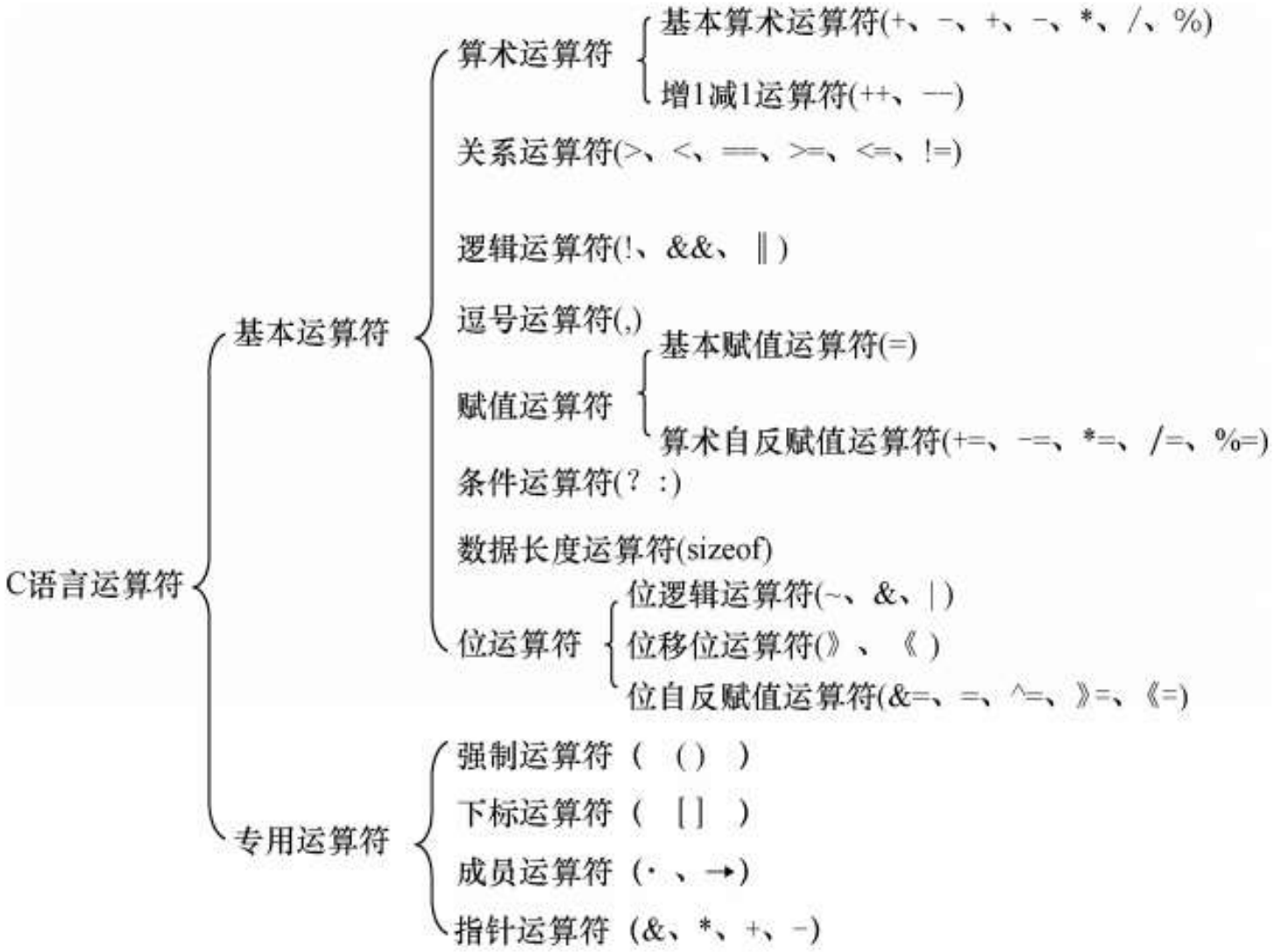


图 2-5 C 语言运算符分类

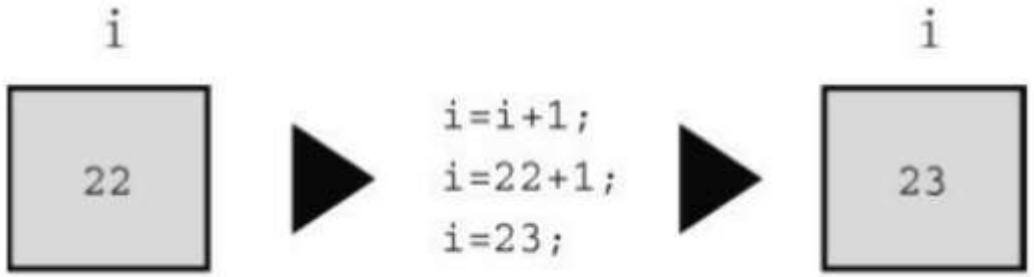
下列少数运算符号有双重含义:

- **+** 在算术运算中既表示“取正”运算， 又可表示“加法”运算；在指针运算中表示“加法”运算。
- **-** 在算术运算中既表示“取负”运算， 又可表示“减法”运算；在指针运算中表示“减法”运算。
- ***** 在算术运算中表示“乘法”运算；在指针运算中表示“指针”运算。
- **&** 在位逻辑运算中表示“与”运算；在指针运算中表示“取地址”运算。

对这些运算符的理解与当时的运算对象有关，在学习时要注意区分。

基本运算符 = + - * /

- =并不意味着“相等”，而是一个赋值运算符。
- 例如：today_data = 20231009;
- =号左侧是一个变量名，右侧是赋给该变量的值。
- 符号=被称为赋值运算符。
- 赋值行为从右往左进行。
- 例如 int I = 22 ; I = I + 1;
- 在数学上不可以，但是在计算机赋值表达式语句中，这很合理。



基本运算符 = + - * /

- 加法运算符（addition operator）用于加法运算，使其两侧的值相加。
- 例如：printf("%d", 4 + 20)打印的是24，而不是表达式4 + 20
- 相加的值（运算对象）可以是变量，也可以是常量。
- age = year_age + month_age;
- 计算机会查看加法运算符右侧的两个变量，把它们相加，然后把和赋给变量age
- 减法运算符（subtraction operator）用于减法运算，使其左侧的数减去右侧的数。
- 例如，下面的语句把95.5赋给score：
- score = 100.0 – 4.5;
- +和-运算符都被称为二元运算符（binary operator），即这些运算符需要两个运算对象才能完成操作。

基本运算符 = + - * /

- 符号*表示乘法。
- 符号/来表示除法。/左侧的值是被除数，右侧的值是除数。
- 整数除法和浮点数除法不同。浮点数除法的结果是浮点数，而整数除法的结果是整数。

```
int main(void)
{
    int i = 5;
    int j = 3;
    int result;
    result = i/j;
    printf("result is %d\n",result);
}
```

```
C:\WINDOWS\system32\cmd.
result is 1
请按任意键继续. . .
```

- 整数除法会截断计算结果的小数部分

求模运算符 %

- 求模运算符（**modulus operator**）用于整数运算。求模运算符给出其左侧整数除以右侧整数的余数

EX02_qiumo.c

(全局范围) main(void)

```
#include <stdio.h>
#define SEC_PER_MIN 60
int main(void)
{
    int sec, min, left;
    printf("Enter the number of seconds:\n");
    scanf("%d",&sec);
    min = sec/ SEC_PER_MIN;
    left = sec % SEC_PER_MIN;
    printf("%d seconds is %d minutes, %d seconds\n",sec,min,left);
    return 0;
}
```

/截断分钟数
%剩下的秒数

```
Enter the number of seconds:
180
180 seconds is 3 minutes, 0 seconds
请按任意键继续. . .
```

```
Enter the number of seconds:
520
520 seconds is 8 minutes, 40 seconds
请按任意键继续. . .
```

递增运算符 ++

- 递增运算符（**increment operator**）执行简单的任务，将其运算对象递增**1**。该运算符以两种方式出现。第**1**种方式，**++**出现在其作用的变量前面，这是前缀模式；第**2**种方式，**++**出现在其作用的变量后面，这是后缀模式。

EX02_Dizeng.c

(全局范围) main(void)

```
#include <stdio.h>
void main(void)
{
    int a = 1, b = 1;
    int a_back, b_forward;
    a_back = a++;
    b_forward = ++b;
    printf("a,a_back,b and b_forward are %d, %d, %d, %d.\n",a,a_back,b,b_forward);
}
```

```
C:\WINDOWS\system32\cmd.
a,a_back,b and b_forward are 2, 1, 2, 2.
请按任意键继续. . .
```

a和b都递增了1，但是，a_back是a递增之前的值，而b_forward是b递增之后的值。

```
(全局范围)                                    main(void)
#include <stdio.h>
void main(void)
{
    int a = 1;
    int q, q_new;
    q = 2 * ++a;
    q_new = 2 * a++;
    printf("a,q and q_new are %d, %d, %d.\n",a,q,q_new);
}
```

```
C:\WINDOWS\system32\cmd.
a,q and q_new are 3, 4, 4.
请按任意键继续. . .
```

前缀形式

```
q = 2*++a;
```

首先，a递增1；
然后，2乘以a，并将结果赋给q

后缀形式

```
q = 2*a++;
```

首先，2乘以a，并将结果赋给q；
然后，a递增1

递减运算符 --

- 递减运算符（decrement operator）用--代替++
- --count; // 前缀形式的递减运算符
- count--; // 后缀形式的递减运算符
- 注意：
- 例如，设有定义语句：int a=2,b=2；则: a+++b 应理解成(a++)+b，先取a值加b作为运算结果（4），后对a加1，a为3。 a---b 应理解成(a--)-b，先取a值减b作为运算结果（0），后对a减1，a为1
- 递增运算符和递减运算符都有很高的结合优先级，只有圆括号的优先级比它们高。
- 例如x*y++表示的是(x)*(y++)，而不是(x*y)++
- 尽量避免复杂的使用：如果一个变量出现在一个函数的多个参数中，不要对该变量使用递增或递减运算符；如果一个变量多次出现在一个表达式中，不要对该变量使用递增或递减运算符。

关系运算符

- C 语言的所有关系运算符

运算符	含义
<	小于
<=	小于或等于
==	等于
>=	大于或等于
>	大于
!=	不等于

- 例如 int m,n; m = (10>2); n = (2 > 10);
- 两个关系表达式的值分别赋给变量m，n
- 表达式为真的值，true = 1, 即m = 1
- 表达式为假的值，false = 0, 即n = 0
- 关系运算符用于构成关系表达式。关系表达式为真时值为1，为假时值为0。通常用关系表达式作为测试条件的语句（如while和if）可以使用任何表达式作为测试条件，非零为真，零为假。

逻辑运算符

- C 语言的三种逻辑运算符

逻辑运算符	含义
&&	与
||	或
!	非

- 逻辑运算符两侧的条件必须都为真，整个表达式才为真。逻辑运算符的优先级比关系运算符低，所以不必在子表达式两侧加圆括号。
- 例如5 > 2 && 4 > 7为假，因为只有一个子表达式为真；
- 5 > 2 || 4 > 7为真，因为有一个子表达式为真；
- !(4 > 7)为真，因为4不大于7。

2.4.2 运算符的优先级和结合性

- 运算符的优先级是指在多个运算符连接了多个运算对象时，运算符与操作数结合次序。
- C语言规定，优先级高的运算符先结合操作数，优先级低的运算符后结合操作数。
- 目前我们学过的运算符优先级如下（由高到低）

运算符	结合律
()	从左往右
+ -（一元）	从右往左
* /	从左往右
+ -（二元）	从左往右
=	从右往左

- 思考：y = 1 * 12 + 2 * 6; y的值是什么呢？

附录二 C语言运算符优先级与结合性

优先级	运算符	功能	适用范围		结合方向
1	( ) [ ] . ->	小括号运算 下标运算 成员运算 指向运算（通过指针存取成员）	表达式 参数表 数组 结构/联合		自左至右
2	! ~ ++ -- + - & * (type) sizeof	逻辑非 按位求反 自增（增1） 自减（减1） 取正、取负 取地址 取内容 强制类型转换 计算占用内存长度	逻辑运算 位运算 自增 自减 算术运算 指针 指针 类型转换 变量/数据类型	单目运算	自右至左
3	* / %	乘 除 整数取模	算术运算		自左至右
4	+ -	加 减			自左至右
6	< <= > >=	小于 小于等于 大于 大于等于	关系运算	双目运算	自左至右
7	== !=	等于 不等于			
11	&&	逻辑与	逻辑运算		自左至右
12	||	逻辑或			
13	?:	条件运算	三目条件运算		自右至左
14	= op=	双目赋值运算、双目复合赋值运算 op 可为下列运算符之一：* / % + - << >> & ^ |			自右至左
15	,	逗号运算（顺序求值）	表达式		自左至右

优先级：括号>单目>算术>关系>逻辑>条件>赋值>逗号 ⇨ 单、算、关、逻、条、赋

结合性：单目、条件、赋值（自右至左）其它（自左至右） ⇨ 单、条、赋（自右至左）

2.4.3 表达式计算中数据类型的自动转换

如果参加运算的两个运算对象的基本数据类型不一致，则按照图2-3的规则进行转换。

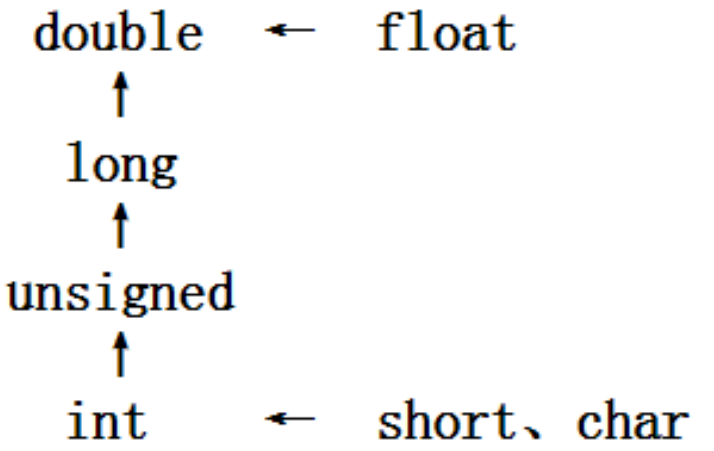


图 2-3 表达式计算时的自动转换规则

“就长不就短”的自动转换规则。

2*16.0+15.0

float

double

double

强制类型转换（显式转换）：(类型说明符) (表达式)

(float) a　　把a转换为实型

(int)(x+y)　　把x+y的结果转换为整型

2.5 算术运算符和表达式

2.5.1基本算术运算符

名称	运算符	对象数与位置	对象类型	运算规则	结果类型	结合性
取正	+	单目前缀	整型 实型 字符型	取原值	与 运算对象 的类型 相同	自右向左
取负	-			取负值		
加	+	双目中缀		相加		自左向右
减	-			相减		
乘	*			相乘		
除	/			相除或整除		
模	%		整型或字符型	整除取余数	整型	

算术运算符的优先级别规定如下:

- 单目基本算术运算符 优先于 双目基本算术运算符。
- *、/、% 优先于 +（加）、-（减）。
- +（取正）、-（取负） 是同级别的，结合性是自右向左的。
- *、/、% 是同级别的，结合性是自左向右的。
- +（加）、-（减） 是同级别的，结合性是自左向右的。

说明:

1、对于除（/）：两整数相除，结果为整数，有一个为实型，结果为实型。要求分母不为0。

2、对于模（%）：要求两侧均为整型数据,余数符号与被除数相同。

例
```
5/2      = 2
-5/2.0 = -2.5
5%2    = 1
5%-2   = 1
-5%2   = -1
-5%-2  = -1
1%-10  = 1
5%1    = 0
5.5%2    (×)
```

2.5.2 增1减1运算符

- ++ --可以前置，也可以后置。
  - » 前置 ++i, --i （先增减后引用）
  - » 后置 i++, i-- （先引用后增减）
  - » i++ ⇔ ++i
- 说明：
  - » ++ -- 不能用于常量和表达式, 如5++, (a+b)++
  - » ++ --结合方向： 自右向左
  - » 优先级：- ++ -- ------>* / % ----->+ -

```
int a, i=1;
a=i++; //表达式值1, a=1, i=2
a=++i; //表达式值2, a=2, i=2
```

应用举例：P35★★★

数学表达式或数学问题	对应的 C 语言表达式
x^3	x*x*x
$2ab$	2*a*b
$\frac{1}{xy}$	1.0/(x*y)
a、b、c 的平均值	(a+b+c)/3.0
求十进制正整数 x 的个位数字	x%10
去掉十进制正整数 x 的个位数字	x/10
设 c 为大写字母，求对应的小写字母	c+32 或 c+'a'-'A'
设 c 为小写字母，求对应的大写字母	c-32 或 c+'A'-'a'

[程序演示]

```
/* Description: A simple of ++ . */
#include <stdio.h>
void main( void )
{
   int j, k;

   j=2;k=++j;        /*j=3 k=3*/
   j=2;k=j++;        /*j=3 k=2*/

   j=2;printf(“%d %d\n”, j++,++j ); /*3 3 printf语句表达式是从右至左执行*/
   j=2;printf("%d %d\n", ++j,j++ ); /*3 2*/

   j=2,k=1;
   printf("%d...", j+++k );
   j=2,k=1;
   printf("%d\n", (j++)+k );/*3...*/
}
```

```
F:\c test\Debug\1.exe
3 3
3 2
3...
3
Press any key to continue
```

```
3 4
4 2
3
3
请按任意键继续. . .
```

VC++6.0后自增运算是要在整条语句结束以后才自加1的！

注意： 不提倡书写复杂的++/ --表达式，不同的编译系统计算出来的结果可能会不一样。

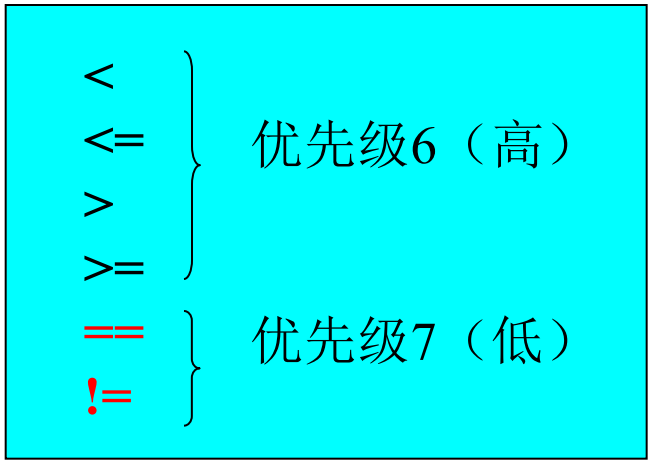
```
void main( void )
{
        int a=1, y;
        y=(++a)+(++a)+(++a);
        printf("%d\n",y);
}
```

VC 10 3+3+4

BC/TC 12 4+4+4

解释： 类似于(++a)+(++a)的表达式有副作用，结果争议性较大，编程中尽量避免; 参见： http://www.cnblogs.com/smwikipedia/articles/1229984.html

2.6 关系运算符和表达式



- 关系运算符
  - 种类：< <= == >= > !=
  - 结合方向：
  - 优先级别：
  - 关系表达式的值：逻辑值“真”或“假”，用1和0表示

例：
```
c>a+b     //c>(a+b)
a>b!=c    //(a>b)!=c
a==b<c    //a==(b<c)
a=b>c     //a=(b>c)
```

例
```
int a=3,b=2,c=1,d,f;
a>b          //1
(a>b)==c     //1
b+c<a        //0
d=a>b        //d=1
f=a>b>c      //f=0
```

例 若a=-5; b=-2; x=-4; 则 a<=x<=b的值为 0
5>2>7>8 值为 0

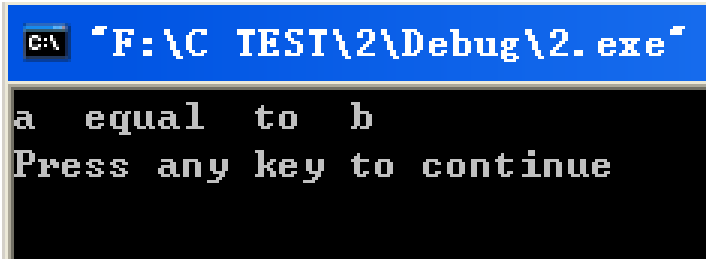
例 应避免对实数作相等或不等的判断
如 1.0/3.0*3.0==1.0 结果为 0
可改写为：fabs(1.0/3.0*3.0-1.0)<1e-6

例 int i=1, j=7,a=i+(j%4!=0); 则a= 2

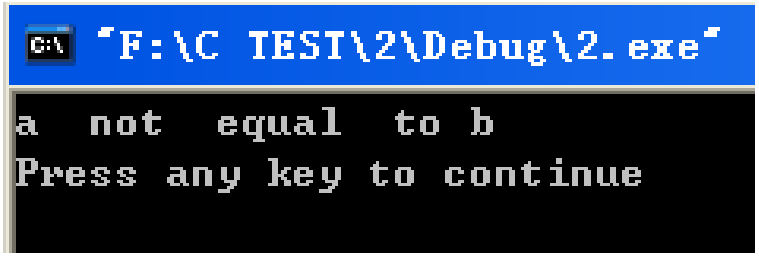
例 ‘a’>0 结果为 1 ‘A’>100 结果为 0

注意区分“=”与“==”

```
#include <stdio.h>
void main( void )
{
int a=0,b=1;
if(a=b)    //a=b结果为1
printf("a  equal  to  b\n");
else
   printf("a  not  equal  to b\n");
}
```




```
#include <stdio.h>
void main( void )
{
int a=0,b=1;
if(a==b)    //a==b结果为0
printf("a  equal  to  b\n");
else
   printf("a  not  equal  to b\n");
}
```



2.7 逻辑运算符和表达式

- 逻辑运算符： ! && ||

●优先级：

!	2	高
&&	11	
||	12	低

●结合方向：

!	←
&&	→
||	→

例 a<=x && x<=b ➔(a<=x) && (x<=b)
a>b&&x>y ➔(a>b)&&(x>y)
a==b||x==y ➔(a==b)||(x==y)
!a||a>b ➔(!a)||(a>b)
5>3&&2||8<4-!0 ➔
(5>3)&&2||(8<(4-(!0))) 值为1

●逻辑表达式求解时，并非所有的逻辑运算都执行，只是在必须执行下一个逻辑运算才能求出表达式的解时，才执行该运算

例 a&&b&&c //只在a为真时，才判别b的值；只在a、b都为真时，才判别 c的值
例 a||b||c //只在a为假时，才判别b的值；只在a、b都为假时，才判别 c的值

[程序演示]

```
#include <stdio.h>

void main( void )
{
   int a=1, b=2, c=3, d=4, m=1, n=1;

        (m=a>b)&& (n=c>d);
       printf(" m=%d n=%d\n",  m, n );
}
```

```
"F:\C IESI\2\Debug\2.exe"
 m=0 n=1
Press any key to continue
```

分析：(m=a>b)的结果为0,0和任意表达式“与”运算均为0，故n=c>d没有运算，n的值仍为1！

想一想，如果这里的表达式是(m=a>b)|| (n=c>d)，结果会怎样？

```
#include <stdio.h>

void main( void )
{
   int a=1, b=2, c=3, d=4, m=1, n=1;

        (m=a>b)|| (n=c>d);
       printf(" m=%d n=%d\n",  m, n );
}
```

```
C:\JMSOFT\CYuYan\bin\wwtemp.exe
m=0 n=0
```

2.8 条件运算符与条件表达式

格式：
e1 ? e2 : e3

优先级：
13（仅比逗号和赋值运算符高）

名称	对象数与位置	运算符	对象类型	运算规则	结果类型	结合性
条件	三目中缀	? :	任何类型的表达式	设有e1?e2:e3 e1为非0, 获得e2 e1为0, 获得e3	e2(或e3)的类型	自右向左 ←

例
```
if (a>b)
    printf("%d",a);
else
    printf("%d",b);
```
⟺
```
printf("%d",a>b?a:b);
```

```
printf("a+|b|=%d\n",b>0?a+b:a-b);
```

```
int  a=2,b=3,c=-1,d;
d=a?b:c;
 //d=(a?b:c),d为3
a=0;
d=a?b:c;  //d为-1
```

```
#include <stdio.h>
void main( void )
{
   int a=1, b=2, c=3, m;
   m=b>a?(b=5):(c=6);   // m=((b>a)?(b=5):(c=6));
   printf(" a=%d b=%d c=%d m=%d\n", a, b, c, m );
}
  //结果:  a=1 b=5 c=3 m=5 ,注意:  c=6这个表达式没有执行
```

2.9 赋值运算符和表达式

- 简单赋值运算：变量标识符=表达式
  - 作用：将一个数据（常量或表达式）赋给一个变量

复合赋值运算：exp1 op= exp2

- 种类：+= -= *= /= %= 《= 》= &= ^= |=
- 含义：exp1 op= exp2 ⇔ exp1 = exp1 op exp2

说明：

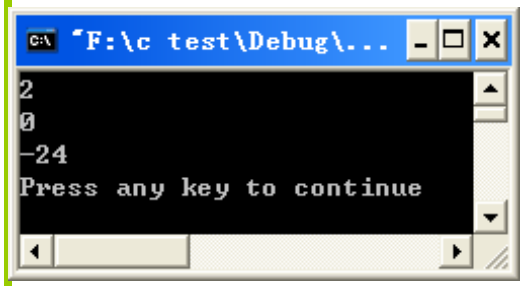
- 结合方向：自右向左,且可嵌套
- 优先级：14
- 左侧必须是变量，不能是常量或表达式
- 赋值转换规则:使赋值号右边表达式值自动转换成其左边变量的类型
- 赋值表达式的值就是被赋值变量的值

例： int i=2.56; //结果i=2;
a=b=c=5 //表达式值为5，a,b,c值为5
a=5+(c=6) //表达式值11，c=6,a=11
a=(b=4)+(c=6) //表达式值10，a=10,b=4,c=6
a=(b=10)/(c=2) //表达式值5，a=5,b=10,c=2

例：

```
#include <stdio.h>
void main( void )
{
    int a=2;
    a%=4;
    printf("%d\n",a);
    a+=a*=a-=a*=3;        // 等价于a=a+(a=a*(a=a-(a=a*3)))
    printf("%d\n",a);

    a=4; a+=a-=a*a;       //等价于a=a+(a=a-(a*a))
    printf("%d\n",a);
}
```



2.10 逗号运算符和表达式

- 形式： 表达式1,表达式2,……表达式n
- 结合性: 从左向右
- 优先级：15（最低）
- 逗号表达式的值：等于表达式n的值
- 用途：常用于循环for语句中

例 a=3*5,a*4 //a=15,表达式值60

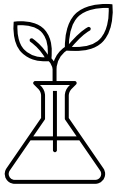
a=3*5,a*4,a+5 //a=15,表达式值20

例 x=(a=3,6*3) //赋值表达式，表达式值18，x=18

x=a=3,6*a //逗号表达式，表达式值18，x=3

填空题 1分

设置



请写出程序运行的结果

```
#include <stdio.h>

void main( void )
{
   int a,x;
   x=(a=3,6*3);
  printf(" x=%d\n",  x );
}
```

[填空1]

作答

小测试

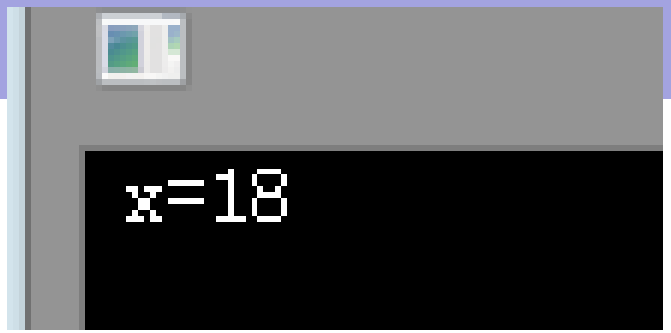
```
#include <stdio.h>

void main( void )
{
   int a,x;

x=(a=3,6*3);

printf(" x=%d\n",  x );

}
```



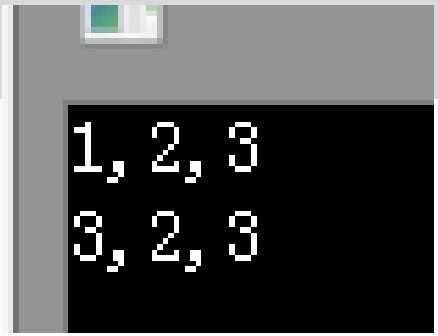

```
#include <stdio.h>

void main( void )
{
   int a=1,b=2,c=3;

printf("%d,%d,%d\n",a,b,c);

printf("%d,%d,%d",(a,b,c),b,c);

}
```



2.11 长度运算符和数据类型转换运算符

1 长度运算符（sizeof） 单目运算符，优先级 2(高)

形式： sizeof 数据对象；
sizeof（类型说明符）；

```
int  a,b,c;
b=sizeof a;  /*b的值为4 */
c=sizeof(float);  /* c的值为4*/
```

【例2-36】长度运算符的使用。

设变量定义如下：char c; int i; short s; long l;
float f; double d;

则：sizeof(c)的值是1 sizeof(i)的值是4
sizeof(s)的值是2 sizeof(l)的值是4
sizeof(f)的值是4 sizeof(d)的值是8
sizeof(unsigned int)的值是4
sizeof(unsigned short)的值是2
sizeof(unsigned long)的值是4

2 数据类型转换运算符（强制类型转换） 单目运算符，优先级 2(高)！

功能：将某个表达式的运算结果转换为另一种数据类型

形式：（数据类型符）（表达式）；例：(int) (a/b)

说明：

(1) 表达式必须加圆括号。

(2) 长度长的数据转换为长度短的数据时，将截取超长的部分，会造成数据值的改变。

【例2-37】数据类型转换运算符的使用。

设变量定义如下：char c=1;　int i=2;　short s=3;　long l=4;
float f=5.0;　double d=6.0;

则：(float)c的值是1.0，单精度实型，而c的值仍然是1，字符型。

(short)l的值是4，带符号的短整型，而变量l仍然是4，带符号长整型。

(int)(s/f)的值是0，带符号整型，变量s和f的类型与值均不改变。

(int)s/f的值是0.6，双精度实型，变量s和f的类型与值均不改变。

(long)(i+d)的值是8，带符号长整型，变量i和d的类型与值均不改变。

填空题 5分

设置

设变量定义如下：
char c=1; int i=2; short s=3; long l=4;
float f=5.0; double d=6.0;
则：
(float)c的值是 [填空1]
(short)l的值是 [填空2]
(int)(s/f)的值是 [填空3]
(int)s/f的值是 [填空4]
(long)(i+d)的值是 [填空5]

作答

2.12 选修 位运算

- 了解位运算的概念和使用方法；
- 了解位运算、位操作；

一、位运算的概念

位运算：以二进制位为单位的运算。

“位运算”仅限于整数（整型数和字符型）。

二、位运算符

位逻辑运算符（~、&、^、|）

位移位运算符（<<、>>）

位复合赋值运算符(&=、|=、^=、<<=、>>=）

2.13 基本数据输入输出

- 在C语言中，数据的输入输出是利用系统函数来实现的。设计人员只要调用相关的系统函数，就可以完成各种数据的输入输出工作。

- 调用系统函数时，需要注意下列几个问题：系统函数的函数名、函数的形式参数及其数据类型、函数的功能、函数调用的返回值、函数调用格式。

- 类比：函数f(x,y)=2x+y，则f(5,8)=2×5+8=18。
  - f称为“函数名”；
  - 2x+y称为“f函数的功能”；
  - x和y称为“f函数的形式参数”，简称“形参”；
  - f(5,8)称为“函数f的调用”；5和8称为函数调用时的“实际参数”，简称“实参”；
  - 18称为“函数调用时的返回值”。
  - 对于任意函数来说，“函数名(实参1，实参2，…)”就是“函数调用格式”。

- 单个字符输入输出函数每次只能输入输出单个字符。
- 而格式化输入输出函数每次能输入输出若干个基本类型的数据，如带符号和不带符号的整型、短整型、长整型、单精度和双精度实型、单个字符、字符串。这种函数不但能输入输出各种基本类型的数据，而且还可以控制输入输出时每个数据的宽度，对实型数据还可以控制小数位数等。
- 注意，凡是使用本节介绍的4个系统函数，必须在程序清单的开始写上下列包含命令：

#include <stdio.h>　或　#include "stdio.h"

2.13.1 字符数据的输入函数

- 该函数的主要功能就是从键盘上读取单个字符，作为函数的返回值。
  - 函数调用格式：getchar()
  - 函数形式参数：无。
  - 函数功能 ：从键盘读取单个字符。
  - 函数返回值 ：读取的单个字符。
- 使用该函数时，一般使用字符型或整型变量来接收该函数的返回值，即键盘上输入的单个字符。通常使用方式是采用下列语句：
- 变量＝getchar();

- 当程序执行中遇到该函数调用时，首先会停止程序的执行，然后等待用户从键盘上输入一个字符作为函数的返回值，再继续执行程序。

- 请读者注意，键盘上输入的任何一个键都是有效的，而且每次输入都需要按“Enter”键（回车换行符）来表示输入的结束。对于一次输入的字符，如果程序中没有使用完，将会自动保留下来，留作下次从键盘上读取数据时使用。

```
char ch;
printf("请输入字符：");//1 2 3 回车
scanf("%c", &ch);// 读取 1
printf("%c", getchar());// 读取并输出 2
printf("%c", getchar());// 读取并输出 3
//printf("%c", getchar());// 读取并输出 回车
```

```
请输入字符：12
2请按任意键继续. . .
```

```
char ch;
printf("请输入字符：");//1 2 3 回车
scanf("%c", &ch);// 读取 1
printf("%c", getchar());// 读取并输出 2
printf("%c", getchar());// 读取并输出 3
printf("%c", getchar());// 读取并输出 回车
```

```
请输入字符：123
23
请按任意键继续. . .
```

【例2-42】字符输入函数的使用。（P48）

- 设程序段如下：**char c1,c2; int i1;**

- **c1=getchar(); i1=getchar();**

- 如果从键盘上输入**1↵ A↵**（书中↵表示回车换行键**Enter**，对应字符‘\n’，下同），则变量**c1**中将获得字符‘**1**’；变量**i1**中将获得字符‘**\n**’的代码值（**13**）。而且，在输完“**1↵**”后，程序将不再要求输入，即“**A↵**”将无法输入，同时没有多余的字符留作下次输入时使用。

- 为了解决这个问题，在程序中经常使用下列方法：

c1=getchar(); c2=getchar();

i1=getchar(); c2=getchar();

  - 当从键盘上输入1↵ A↵ 后，则字符‘1’、‘A’依次存入c1、i1，而c2中获得字符‘\n’。

- 也可以使用下列方法：

c1=getchar(); i1=getchar(); c2=getchar();

  - 此时，从键盘上输入1A↵，则变量c1中将获得字符'1'；变量i1中获得字符'A'的代码值65；而c2中获得字符'\n'。

2.13.2 字符数据的输出函数

- 该函数的主要功能就是将字符型常量、字符型变量或字符型表达式值对应的单个字符输出到显示器显示。
  - 函数调用格式：putchar(c)
  - 函数形式参数：c是字符常量、字符变量、字符型表达式、整型表达式。
  - 函数功能　：将参数c对应的字符输出到显示器上。
  - 函数返回值　：c对应的字符。
- 该函数的一般使用方式是利用下列语句：

  putchar(表达式);

  - 其中的表达式最终结果应是某个字符。

【例2-43】字符输出函数的使用。

- 设程序段如下：

```
#include <stdio.h>
void main( void )
{
  char c1='1';  int i1=65;
  putchar(c1);  putchar('\n');
  putchar(i1); putchar(c1+17);
}
```

结果分析：

1　（在字符1的后面有一个回车换行符）

AB　(为什么输出c1+17为字母“B”）

2.13.3 格式化输出——printf()函数

本函数能够将若干个各种基本类型的数据送到显示器上显示，并且可以控制显示时数据的类型（即整型、实型、字符型、字符串）、数据的宽度（即输出字符的个数）、实数的小数位。

函数调用格式： printf(输出格式字符串,输出表达式表)

函数形式参数： 输出格式字符串是由控制输出格式的格式字符等组成的字符串。输出表达式表是用逗号分隔的若干个表达式。

函数功能　　： 依次计算“输出表达式表”中诸表达式的值，然后按照“输出格式字符串”中对应的格式字符规定的格式输出到显示器上。

函数返回值 ： 输出数据的个数。

格式字符、对应的数据类型、数据的输出形式和数据输出格式如表2-20所示。

表2-20 输出格式字符表

格式字符	对应数据类型	数据输出形式	数据输出格式
%-md	int short unsigned int unsigned short	十进制整数	无m：按数据的实际宽度位数输出 有m：输出m位， 数据超过m位，按实际宽度输出 数据不足m位，按下列规则补空格 ： 无-号 右对齐(左补空格) 有-号 左对齐(右补空格) n为输出浮点数精度
%-mo		八进制整数	
%-mx		十六进制整数	
%-mu		无符号十进制整数	
%-mld	long unsigned long	十进制整数	
%-mlo		八进制整数	
%-mlx		十六进制整数	
%-mlu		无符号十进制整数	
%-m.nf	float double	十进制小数	
%-m.ne		十进制指数	
%g		自动选f或e中宽度小的格式	
%-mc	char	单个字符	无m：输出单个字符 有m：输出m位，按下列规则补空格 无-号 右对齐 有-号 左对齐
%-m.ns	字符串	一串字符	无m.n：按实际字符串输出全部字符 有m.n：输出前n个字符，按下列规则 补空格： 无-号 右对齐 有-号 左对齐

注：表2-20中的“m、n”分别是整型常量，可以省略。

2. 输出表达式表

使用输出表达式表是若干个、用“逗号”分隔的表达式组成的。特别要注意的是,这些表达式虽然用“逗号”分隔，但不是“逗号表达式”。
使用格式化输出函数时，需要注意下列几点：

（1）对输出的数据，如果数据不超出范围，则整型和字符型可以通用。

（2）其中的m或n是一个整型常量，m主要用来控制输出数据的宽度，n用来控制小数位数（对实型数据）或字符串的实际输出字符数（对字符串数据）。m和n可以省略，省略时按数据的实际宽度输出。

（3）在两个输出数据之间不会自动增加一个空格符；
例如，“printf("%d%d%d",12,3456,789);”的输出结果将是123456789。
在所有数据输出之后，也不会自动增加一个回车换行符。
例如，“printf("%d",12); printf("%d",3456);”的输出结果将是123456。

（4）如果一次要输出多个数据，这些数据可以用下列三种方法来分隔：

设有数据定义语句：int a=12,b=3456,c=789;

- 方法一 选用合适的宽度m来分隔。

例如，语句如下：printf("%6d%6d%6d\n",a,b,c);

则输出结果如下：□□□□12□□3456□□□789←┘

- 方法二 用非格式字符作为分隔符号(常用逗号)。

例如，语句如下：printf("%d,%d,%d\n",a,b,c);

输出结果如下：12,3456,789←┘

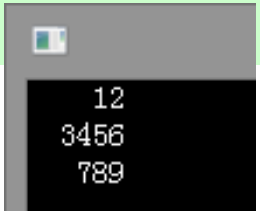
- 方法三 用非格式字符作为分隔符号(常用变量名=)。

例如，语句如下：printf("a=%d b=%d c=%d\n",a,b,c);

输出结果如下：a=12 b=3456 c=789←┘

示例：

```
#include <stdio.h>
void main( void )
{
int a=12,b=3456,c=789;
printf("%6d\n%6d\n%6d",a,b,c
);
}
```



（5）程序中常用的输出格式：

带符号整型数据	%d	不带符号整型数据	%u
带符号长整型数据	%ld	不带符号长整型数据	%lu
实型数据	%m.nf	字符型数据	%c
字符串数据	%s		

【例2-45】格式化输出函数的使用。

设程序段如下：char c1='1',c2='A';int i1=-2; float f1=-123.456;

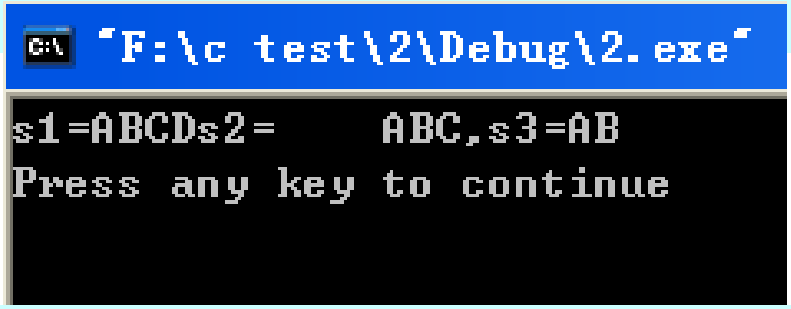
printf("f1=%f,%8.2f,%-8.1f,%e,%g\n",f1,f1,f1,f1,f1);
输出结果：
f1=-123.456001,□-123.46,-123.5□□,-1.234560e+002,-123.456↵

printf("c1=%d,c2+2=%3c%-3c\n",c1,c2+2,'a');
输出结果：c1=49,c2+2=□□C,a□□↵

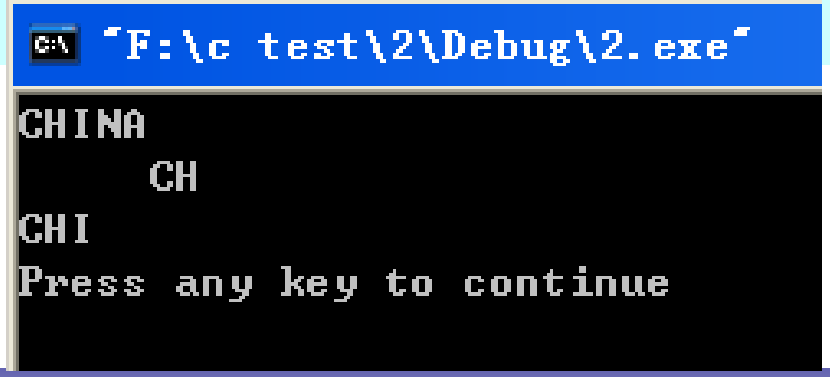
注意：%-m.nf，无-号，右对齐，左补空格（有-号，左对齐，右补空格），m为总宽度，n为精度！

示例（字符串输出）

```
#include <stdio.h>
void main( void )
{
printf("s1=%ss2=%7.3s,s3=%-7.2s\n","ABCD","ABCD","ABCD");
}
```




```
#include <stdio.h>
void main( void )
{
printf("%s\n%7.2s\n%-5.3s\n", "CHINA", "CHINA","CHINA");
}
```



2.13.4 格式化输入——scanf()函数

本函数能够从键盘上输入各种基本类型的数据，并且可以控制输入时数据的类型（即整型、实型、字符型、字符串）、数据的宽度（即输入字符的个数）。

- 函数调用格式：scanf(输入格式字符串,输入变量地址表)
- 函数形式参数：输入格式字符串是由控制输入格式的格式字符等组成的字符串。输入变量地址表是用逗号分隔的若干个接收输入数据的变量地址。
- 函数功能 ：按照“输入格式字符串”中规定的输入格式，从键盘上读取若干个数据按“输入变量地址表”中变量自左向右的顺序，依次存入对应的变量。
- 函数返回值 ：从键盘上读取数据的个数。

下面解释该函数调用时的两个参数。

1. 输入格式字符串

- 输入格式字符串是由“格式字符”和“非格式字符”组成的，通常是一个字符串常量。
- “非格式字符”必须原样原位置输入。可以没有，通常是作为输入时数据的间隔符号，常用的是“逗号”。
- “格式字符”分为整型、长整型、单精度实型、双精度实型、字符型、字符串型。每个格式字符对应一个要输入的数据，从键盘上输入数据时，必须严格按照规定的格式输入。
- 格式字符、对应的数据类型、数据的输入形式和数据输入格式如表2-19所示。

表2-19 输入格式字符表

格式字符	对应数据类型	数据输入形式	数据输入格式
%md	int short unsigned int unsigned short	十进制整数	无m：按数据的实际宽度输入 有m：只取输入数据的前m位， 不足m位则需要后跟Enter键
%mo		八进制整数	
%mx		十六进制整数	
%mld	long unsigned long	十进制整数	
%mlo		八进制整数	
%mlx		十六进制整数	
%mf %me	float	十进制小数 十进制指数	
%mlf %mle	double	十进制小数 十进制指数	
%mc	char	单个字符	无m:仅取单个字符 有m:只取输入的m位中第一个字符
%ms	字符串	字符串	无m:取若干字符直到回车或空格为止 有m:仅取输入的前m个字符

注：表2-19中的“m”是一个整型常量，可以省略。

2. 输入变量地址表

- 输入变量地址表是由接受输入数据的变量地址组成的，变量地址之间用“逗号”分隔。
- 变量的地址必须写成“&变量名”。在基本数据类型中，只有字符串没有对应的字符串变量，如何接受输入的字符串将在第4章（数组）中介绍。

使用格式化输入函数时，需要注意下列几点：

- （1）对接受数据的变量，如果数据不超出范围，则整型和字符型是可以通用的。
- （2）所需数据从键盘上输入结束后，要跟一个Enter键表示输入结束。
- （3）用%c作为输入格式字符时，作用和“getchar()”完全相同，仅接受单个字符。由于每次输入都需要用Enter键结束，为了避免多余字符留作下次使用，可以采用“%2c”的格式，输入形式为“字符←┘”。

- （4）如果一次要输入多个数据，这些数据可以用下列三种方法来分隔：假定，要使整型变量a、b、c分别获得12、3456、789。

◆ 方法一 选用合适的宽度m来分隔。
- 例如，语句如下：
- scanf("%2d%4d%3d",&a,&b,&c);
- 从键盘上只要输入：123456789↵

```
#include <stdio.h>
void main( void )
{
 int a,b,c;
 scanf("%2d%4d%3d",&a,&b,&c);
 printf("%d\n%d\n%d\n",a,b,c);
}
```

◆ 方法二 用系统规定的分隔符号(空格符、Tab符、回车换行符)。
- 例如，语句如下：scanf("%d%d%d",&a,&b,&c);
- 从键盘上只要输入：12□3456□789↵（其中的□表示空格符）
- 或者输入：12 Tab符3456 Tab符789↵
- 或者输入：12 ↵ 3456 ↵ 789↵

◆ 方法三 用非格式字符作为分隔符号(常用逗号)。
- 例如，语句如下：scanf("%d,%d,%d",&a,&b,&c);
- 从键盘上只要输入：12,3456,789↵

- （5）程序中常用的输入格式如下：

整型数据	%d	长整型数据	%ld
单精度实型数据	%f	双精度实型数据	%lf
字符型数据	%c	字符串数据	%s

- 使用上述格式，在数据输入时，整型、实型数据用非格式字符“逗号”分隔；字符型数据不分隔；字符串型数据用空格或←┘分隔。

【例2-44】格式化输入函数的使用。

设变量定义如下：char c1,c2,c3;
int i1,i2,i3,i4; long l1,l2,l3,l4;
float f1,f2,f3,f4; double d1,d2,d3,d4;

- 设有函数调用：scanf("%d,%d",&i1,&i2);
  要使变量i1的值为-2、i2的值为288，则必须输入-2,288↵ 。

- 设有函数调用：scanf("%lf,%lf",&d1,&d2);
  要使变量d1的值为-2.0、d2的值为0.088，则可以输入-2.0,0.088↵ 。

- 设有函数调用：scanf("%c%c",&c1,&c2);
  要使变量c1的值为'B'、c2的值为'9'，则必须输入B9↵ 。

- 设有函数调用：scanf("%c%4d%6ld%5f%6lf",&c3,&i3,&l3,&f3,&d3);
  要使变量c3、i3、l3、f3、d3的值依次为'1'、2、3L、4.0、5.0，输入格式如下：
  1□□□2□□□□□3□□4.0□□□5.0↵ （其中的□表示空格符）

- 设有函数调用：scanf("%d%ld%f%lf",&i4,&l4,&f4,&d4);
  要使变量i4、l4、f4、d4的值依次为1、2L、3.0、4.0，则输入格式如下：
  1□2□3.0□4.0↵ （其中的□表示空格符,作为输入数据的间隔符）

示例

例 scanf(“%d%o%x”,&a,&b,&c);
printf(“a=%d,b=%d,c=%d\n”,a,b,c);
输入 123 123 123↵
输出 a=123,b=83,c=291

例 scanf(“%d,%d”,&a,&b)
输入 3,4 ↵
则3⇒a, 4 ⇒b

例 int a,b,c;
scanf("a=%d,b=%d,c=%d",&a,&b,&c);
printf("%d\n%d\n%d\n",a,b,c);
输入 a=12,b=24,c=36 ↵
//（此时a=、b=、c=应原样输入）