高级语言程序设计

主讲：王老师

尚德机构

学习是一种信仰

第三章　数据类型、运算符和表达式

第一节 数据类型

数据既是程序加工和处理的对象，也是程序加工和处理的结果。

1、基本类型包括整型、实型（又称浮点型）、字符型和枚举型四种。

2、构造类型是由基本数据类型按一定规则组合而成的复杂数据类型，包括数组型、结构体类型和共用体类型。

3、指针类型是一种更加灵活的数据类型，用来表示内存地址。

4、空类型是从语法完整性角度给出的数据类型，表示不需要数据值，因而没有类型。

数据类型决定：

1. 数据占内存字节数
2. 数据取值范围
3. 可以进行的操作

第二节 常量

常量又称为常数，是在程序运行过程中其值不能改变的数据。

常量-整型常量

整型常量：

整型常量通常称为整数，不带有小数部分，包括正整数、负整数和0。

C语言中，整型常量有三种书写形式：

1、八进制整型常量，必须以0开头，即以0作为八进制整数的前缀，是由数字0~7组成的数字串。

2、十进制整型常量，即通常的整数，没有前缀，是由数字0~9组成的数字串，是C语言中用的最多的一种形式。

3、十六进制整型常量，必须以0X或0x开头，即以0X或0x作为十六进制整数的前缀，是由字符0~9、a~f或A~F组成的字符串。

常量-实型常量

实型常量又称为浮点型常量或实数，实型常量只使用十进制。

C语言的实型常量有两种书写形式：

1、十进制小数形式，由整数部分、小数点和小数部分组成，其中小数点必须有，整数部分和小数部分可以省略其中一个，正、负通过前面的“+”（可以省略）和“-”区分。

2、指数形式（用E或e表示），由尾数部分、阶码标志（小写字母e或大写字母E）和指数部分组成，其一般形式为aEb或aeb，其中尾数部分a的位数决定了数值的精度，指数部分b的位数决定了可表示的数值范围。a和b均有正负之分，且b必须为整数。

常量-字符常量

字符常量包括普通字符常量和转义字符常量两种。

普通字符常量是用一对单引号括起来的单个字符，例如‘a’、‘？’均为普通字符常量。C语言还允许使用一种特殊形式的字符常量，即以反斜杠“\”开头的转义字符，它只代表某一特定的ASCII码字符。

C语言规定字符常量中的字母是区分大小写的。

每个字符常量在内存中占用1字节，具体存放的是该字符对应的ASCII码值。

需要注意的是，C语言中空字符和空格符号不同。空字符用转义字符“\0”表示，其值是0；空格符号是ASCII字符集中的一个可显示字符，其ASCII值是32，在屏幕上显示为空格。

常量-字符常量

表 3-1　常用的转义字符常量

转义字符	含　义	转义字符	含　义
\n	回车换行	\t	横向跳到下一个制表位置
\v	垂直制表	\b	左退一格
\r	回车	\f	换页
\0	空字符	\\	反斜杠
\'	单引号	\"	双引号
\ddd	1～3 位八进制数所代表的字符，例如'\103'代表字符'C'	\xhh 或\Xhh	1～2 位十六进制数所代表的字符，例如'\x46'代表字符'F'

需要注意的是，C语言中空字符和空格符号不同。空字符用转义字符“\0”表示，其值是0；空格符号是ASCII字符集中的一个可显示字符，其ASCII值是32，在屏幕上显示空格。

常量-字符串常量

字符串常量又称为“字符串”，是用双引号括起来的零个或多个字符的序列。字符串中字符的个数称为字符串长度。

编译时自动在字符串的末尾处加一个转义字符‘\0’作为字符串的结束标志，即长度为n个字符的字符串占用n+1字节。

字符常量与字符串常量是不同的，其区别在于：

1、定界符不同，字符常量使用单引号，而字符串常量使用双引号。

2、长度不同，字符常量的长度固定为1；而字符串常量的长度，可以是0，也可以是某个整数。

3、存储要求不同，字符常量存储的是字符的ASCII码值，在内存中只占用**1**字节；而字符串常量，除了要存储有效的字符外，还要存储一个字符串结束标志‘\0’。

★字符串常量

- ❖定义：用双引号(“ ”)括起来的字符序列

  “How do you do”，“CHINA”，“a”，“$123.45”

- ❖存储：每个字符串尾自动加一个 ‘\0’ 作为字符串结束标志

例　字符串“hello”在内存中

例　空串 “ ”

- ❖字符常量与字符串常量不同

例　‘a’ 

　　“a” 

例：
```
char  ch;
ch='A';
```

常量-符号常量

符号常量使用之前必须先定义，其定义的一般格式：

#define 符号常量 常量

其中，常量可以是任意类型。符号常量一般采用大写字母表示，而变量一般采用小写字母表示。符号常量的定义一般放在程序的开头，每个定义必须独占一行，因为不是语句，所以后面没有分号“；”。

使用符号常量的优势在于：

1、提高程序的可读性

2、便于程序修改

小结

★常量和符号常量

- ❖定义：程序运行过程中，其值不能被改变的量（常数）
- ❖分类：直接常量、符号常量

类型	示例
整型常量	12 、 0 、 -3
实型常量	4.6 、 -1.23
字符常量	‘a’ 、 ‘b’
符号常量	PRICE 、 PAI

- **符号常量：用标识符代表常量**
  - **一般用大写字母：　PRICE 、 PI**
  - **定义格式：　#define　符号常量　常量**
  - **其值在作用域内不能改变和再赋值。**

```
例 符号常量举例
#define   PRICE   30
#include <stdio.h>
void main()
{
    int num,total;
    num=10;
    total=num*PRICE;
    printf("total=%d\n",total);
}
```

运行结果：

```
total=300
```

符号常量的优点是：见名知意、一改全改

课堂练习

1、正确的C语言整型常数是（　　）

A. 3

B. 3.0

C. 1E0.5

D. false

课堂练习

1、正确的C语言整型常数是（　　）

A. 3

B. 3.0

C. 1E0.5

D. false

答案：A

解析：C语言中，整型常量有三种形式：1.八进制整型常量，必须以0开头，即以0作为八进制整数的前缀，是由数字0~7组成的字符串。2.十进制整型常量，即通常的整数，没有前缀，是由数字0~9组成的字符串。3.十六进制整型常量，必须以0X或0x开头，即以0X或0x作为十六进制整数的前缀，是由字符0~9、a~f或A~F组成的字符串。

课堂练习

2、不属于字符型常量的是（　　）

A. "s"

B. '8'

C. 'A'

D. '\n'

课堂练习

2、不属于字符型常量的是（　　）

A. "s"

B. '8'

C. 'A'

D. '\n'

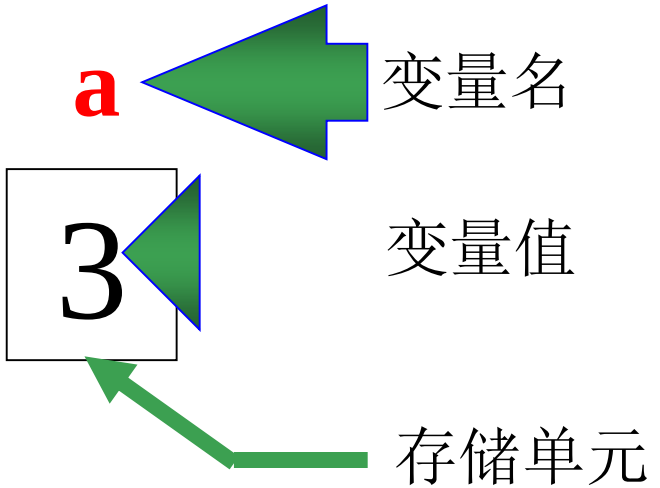
答案：A

解析：字符常量包括普通字符常量和转义字符常量两种，普通字符常量是用一对单引号括起来的单个字符，例如 'a' 、 '？' 均为普通字符常量。C语言还允许使用一种特殊形式的字符常量，即以反斜杠"\"开头的转义字符，它只代表某一特定的ASCⅡ码字符。

第三节　变量

- 定义：其值可以改变的量。
- 定义格式：**数据类型　变量名；**
- 变量应该有名字，并在内存中占据一定的存储单元。
- 变量名和变量值有不同的含义
  - 变量名实为一个符号地址

```
例　变量的使用
main()
{
    int a;
    a=3;
    printf("a=%d",a);
}
```



变量-整型变量

整型变量分为四种：

1、基本整型，类型关键字为int。

2、短整型，类型关键字为short[int]。

3、长整型，类型关键字为long[int]。

4、无符号整型，又称为无符号基本整型（unsigned [int]）、无符号短整型（unsigned short）和无符号长整型（unsigned long）三种。

在16位编译器中，一般一个int型变量占用2字节，且long型（4字节）≥int型（2字节）≥short型（2字节）；而在32位编译器中，一个int型变量占用4字节，long型变量占用4字节，short型变量占用2字节。

变量-实型变量

实型变量分为两种：

1、单精度实型，类型关键字为float，一般占用内存4字节，保留6~7位有效数字。

2、双精度实型，类型关键字为double，一般占用内存8字节，保留15~16位有效数字。

例如：float a;

a=123456.789　//实际a的值是 123456.7

double b;

a=123456.789　//实际b的值是 123456.789

变量-字符型变量

字符型变量的类型关键字是char，存放的是字符的ASCII码值（无符号整数），占用内存1字节。

需要注意的是，字符串只能是常量，C语言中没有字符串变量，而是用字符数组存放字符串变量。

表 3-2　变量的类型及其含义

数据类型	数据类型符	占用字节数/字节	数值范围
基本整型	int	2（或 4）	同短整型（或长整型）
短整型	short [int]	2	-32768～32767（-2^{15}～$2^{15}-1$）
长整型	long [int]	4	-2 147 483 648～2 147 483 647（-2^{31}～$2^{31}-1$）
无符号基本整型	unsigned [int]	2（或 4）	同无符号短整型（或无符号长整型）
无符号短整型	unsigned short	2	0～65535（0～$2^{16}-1$）
无符号长整型	unsigned long	4	0～4 294 967 295（0～$2^{32}-1$）
单精度实型	float	4	-3.4×10^{38}～3.4×10^{38}（保留 7 位有效数字）
双精度实型	double	8	-1.7×10^{308}～1.7×10^{308}（保留 15 位有效数字）
字符型	char	1	-128～127

变量的定义

C语言规定，任何变量**必须先定义后使用**。变量定义语句的一般格式：

数据类型符 变量名1[,变量名2,...];

对变量进行定义时，需要注意以下几个方面：

1、变量定义可以存放在函数之外，也可以放在函数体或符合语句中。

2、被定义为整型（包括int、short和long）的变量，若其值为-128~127，则可以作为字符型变量使用。

3、被定为无符号整型（包括unsigned、unsigned short和unsigned long）的变量，若其值0~255，也可以作为字符型变量使用。

变量的定义

4、被定义为字符型的变量，可以作为整型（包括int、short和long）变量使用，其将值为-128~127，也可以作为无符号整型（包括unsigned、unsigned short和unsigned long）变量使用，其值将为0~255。

5、变量定义后，系统自动为其分配连续的内存单元，所占用的内存字节数取决于变量的数据类型。

变量的初始化

变量的初始化是给变量赋值的一种方式，定义变量的同时给变量赋初值称为变量的初始化。

变量初始化语句的一般格式：

数据类型符 变量名1=初值1[，变量名2[=初值2],...];

```
int  radius=2，area;
short m=1，n=2;
long l1=123L，l2;
```

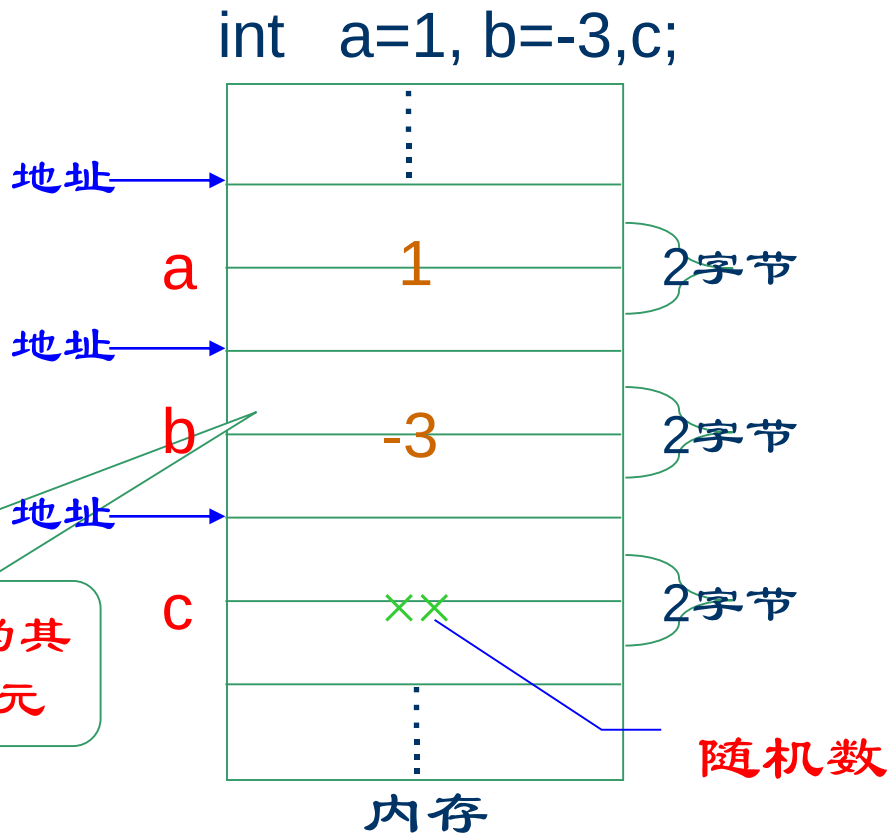
变量赋初值

- 变量的使用：先定义，后使用
- 变量定义位置：一般放在函数开头
- 变量初始化：可以在定义时赋初值

错！ int a=b=c=3;

例：

```
int  a=1,b= -3,c;
float  data=3.67;
char  ch='A';
int  x=1,y=1,z=1;
int  x=y=1;（×）
```



编译程序根据变量定义为其分配指定字节的内存单元

有名常量的定义

C语言中，如果定义了变量并赋予其初值，但不希望在程序中对其值进行修改，则可以将该变量定义为有名常量。

有名常量定义语句的一般格式：

const 数据类型符 变量名1=初值1[,变量名2=初值2,...];

const int j=1,k=2;

const char ch1= ‘y’ ,ch2= ‘N’ ;

课堂练习

1、一个字符型变量所占内存的字节数是_____。

课堂练习

1、一个字符型变量所占内存的字节数是_____。

答案：1

解析：字符型变量的类型关键字是char，存放的是字符的ASCII码值，占用内存1字节。

课堂练习

2、设 char ch；下列语句错误的是（　　）

A. ch= “a=b” ;

B. ch= ‘\0’ ;

C. ch= ‘7’ + ‘9’ ;

D. ch=7+9;

课堂练习

2、设 char ch；下列语句错误的是（　　）

A. ch=“a=b”；

B. ch=‘\0’；

C. ch=‘7’+‘9’；

D. ch=7+9;

答案：A

解析：字符型变量的类型关键字是char，存放的是字符的ASCII码值。

第四节 运算符和表达式

运算符分类：

运算符是进行各种运算的符号，例如常用的加、减、乘、除符号，即是运算符。

表 3-3 C 语言运算符的分类

基本运算符	算术运算符	+（加）、-（减）、*、/、%、+（取正）、-（取负）
	自增、自减运算符	++、--
	关系运算符	>、>=、<、<=、==、!=
	位运算符	位逻辑运算符~、&、|、^ 位移位运算符<<、>>
	逻辑运算符	&&、||、!
	赋值运算符	基本赋值运算符 = 复合赋值运算符 +=、-=、*=、/=、%=、&=、|=、^=、>>=、<<=
	条件运算符	?:
	逗号运算符	,
	数据长度运算符	sizeof
专用运算符	强制类型转换运算符	()
	成员运算符	->、.
	下标运算符	[]
	指针运算符	&、*

运算符

- 算术运算符：（+ - * / % +（取正） -（取负） ）
- 自增、自减运算符 ++ --
- 关系运算符：（< <= == > >= !=）
- 逻辑运算符：（（! && ||）
- 位运算符 ：（<< >> ~ | ^ &）
- 赋值运算符：（= += -= *= /=等）
- 条件运算符：（?:）
- 逗号运算符：（,）
- 数据长度运算符：（sizeof）

- 强制类型转换：（类型）
- 成员运算符：（. ->）
- 下标运算符：（[]）
- 指针运算符：（* &）

★学习运算符应注意：

- 运算符功能
- 与运算量关系
  - 要求运算量个数
  - 要求运算量类型
- 运算符优先级别
- 结合方向
- 结果的类型

结合性

结合性是C语言的独有概念。除单目运算符、赋值运算符和条件运算符是右结合之外，其他的运算符都是左结合。

表 3-4　C 语言运算符的优先级和结合性

优先级	运算符	结合性
1	() [] -> .	从左至右
2	! ~ ++ -- *（取内容） &（取地址） +（取正） -（取负） sizeof	从右至左
3	*（乘） / %	从左至右
4	+（加） -（减）	
5	<< >>	
6	< <= > >=	
7	== !=	
8	&（位逻辑与）	
9	^	
10	|	
11	&&	
12	||	
13	?:	从右至左
14	= += -= *= /= %= &= ^= |= >>= <<=	
15	,	从左至右

算术运算符

C语言的算术运算符包括+、-、*、/、%、+（取正）和-（取负）。

名称	运算符	运算对象个数	功能	运算对象类型	结果类型	结合性
正	+	单目前缀	取正	整型或实型	整型或实型	从右至左
负	-		取负			
加	+	双目中缀	加法			从左至右
减	-		减法			
乘	*		乘法			
除	/		除法			
模	%		整除取余	整型	整型	

算术运算符

1、单目运算符+、-的优先级相同，结合性是从右至左。

2、双目运算符*、/、%的优先级相同，结合性是从左至右。

3、双目运算符+、-的优先级相同，结合性是从左至右。

4、单目算术运算符优先于双目算术运算符。

5、*、/、%优先于+、-。

需要注意的是：

1、除法运算的结果和运算对象的数据类型有关，若两个整数相除，则运算称为整除，其商为整数，小数部分被舍弃。

2、模运算，要求两侧的运算对象必须为整型，否则出错。

★算术运算符和算术表达式

❖基本算术运算符：　+ − * / %

- 结合方向：从左向右
- 优先级：　− ----->* / % -----> + −
  (2)　(3)　(4)

说明：

- “−”可为**单目**运算符时，**右结合性**
- 两整数相除，结果为整数
- %要求两侧均为整型数据
- + − * / 运算的两个数中有一个数为实数，结果是double型

例　5/2　= 2
　　-5/2.0 = -2.5

例　5%2　= 1
　　-5%2　= -1
　　1%10　= 1
　　5%1　= 0
　　5.5%2　(×)

自增、自减运算符

自增和自减运算符分别为++和--，它们都是单目运算符，用于对整型、字符型和指针型变量的值加1和减1的运算，运算结果仍是原来的数据类型。

名称	运算符	运算对象个数	表达式实例	功能描述	运算对象类型	结果类型	结合性
自增	++	单目前缀	++a	先将 a 加 1，然后再引用	整型、字符型、指针型等变量	同运算对象的类型	从右至左
		单目后缀	a++	先引用，然后将 a 加 1			
自减	--	单目前缀	--a	先将 a 减 1，然后再引用			
		单目后缀	a--	先引用，然后将 a 减 1			

自增、自减运算符

1、运算符放在变量之前，即++变量、--变量，则先使变量的值加（或减1），然后再以变化后的值参与其他运算，即先加（或减）1、后运算。

2、运算符放在变量之后，即变量++、变量--，则变量先参与其他运算，然后再使变量值加（或减）1，即先运算、后加（或减）1。

❖自增、自减运算符++ - -

- 作用：使变量值加1或减1
- 种类：
  - ◆前置 ++i, --i （先执行i+1或i-1，再使用i值）
  - ◆后置 i++, i-- （先使用i值，再执行i+1或i-1）

例

```
j=3;  k=++j;                  //k=4, j=4
j=3;  k=j++;                  //k=3, j=4
j=3;  printf("%d",++j);      //4, j=4
j=3;  printf("%d", j++);     //3, j=4
a=3;b=5;c=(++a)*b;            //c=20, a=4
a=3;b=5;c=(a++)*b;            //c=15, a=4
```

- **几点说明：**
  - ++ -- **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：** - ++ -- ------>* / % ----->+ -
    (2) (3) (4)
  - **该运算符常用于循环语句中，使循环变量加减1**

例 -i++ ⇔ -(i++)

```
i=3;  printf("%d",-i++); //-3
```

❖有关表达式使用中的问题说明

- **不同系统对运算符和表达式的处理次序不同，尽可能写通用性强的语句**
- **不要写有歧义和不知系统如何执行的程序**

例如： int x=10,y=5; x++,x+y++**的值是** 16

自增、自减运算符

使用自增和自减运算符时，需要注意以下几点：

1、自增、自减运算，常用于循环语句中，使循环控制变量加（或减）1，以及指针变量中，使指针向下（或上）一个地址。

2、自增、自减运算符，不能用于常量和表达式。

3、在表达式中连续使同一个变量进行自增或自减运算时，很容易出错，所以最好避免这种用法。

自增和自减运算符的优先级如下：

1、自增和自减运算符优先于双目算术运算符；

2、自增、自减运算符和单目算术运算符+、-的优先级相同，结合性是从右至左。

算术表达式

表达式中的运算符都是算术运算符的称为算术表达式，通常算术表达式由运算对象（常量、变量和函数等）、圆括号和算术运算符组成。

算术表达式的构成规则如下：

1、数值型常量、数值型变量和数值型函数调用是算术表达式。

2、+（算术表达式）、-（算术表达式）是算术表达式。

3、++整型变量、--整型变量、整型变量++、整型变量--是算术表达式。

4、（算术表达式）双目算术运算符（算术表达式）是算术表达式。

5、有限次地使用上述规则获得的表达式都是算术表达式。

关系运算符

关系运算符包括>、>=、<、<=、==和！=六种，它们是双目运算符。本质上，关系运算符是比较运算，即两个数据进行比较，判断两个数据是否符合给定的关系。

如果关系运算的结果是逻辑值“真”（即关系成立），用整数“1”表示；如果关系运算的结果是逻辑值“假”（即关系不成立），用整数“0”表示。

名称	运算符	运算对象个数	运算规则	运算对象类型	结果类型	结合性
小于	<	双目中缀	关系成立则为真，结果为“1”；否则为假，结果为“0”	整型或实型或字符型	0或1（整型）	从左至右
小于或等于	<=					
大于	>					
大于或等于	>=					
等于	==					
不等于	!=					

关系运算符

关系运算符的优先级如下：

1、算术运算符优先于关系运算符。

2、>、>=、<、<=优先于==、！=。

3、>、>=、<、<=的优先级相同，结合性是从左至右。

4、==、！=的优先级相同，结合性是从左至右。

需要注意的是，C语言中“等于”这一关系运算符是双等号“==”，而不是单等号“=”（赋值运算符）。

关系表达式

关系表达式是用关系运算符将两个表达式连接起来的表达式。

关系表达式的一般形式：表达式 关系运算符 表达式

表达式主要是算术表达式，也可以是字符型数据或关系表达式、逻辑表达式、条件表达式、赋值表达式、逗号表达式等。由于条件、赋值、逗号运算符的优先级低于关系运算符，所以应注意加圆括号。

逻辑运算符

逻辑运算符包括&&、||和！三种。

名称	运算符	运算对象个数	运算规则	运算对象类型	结果类型	结合性
非	!	单目前缀	见表 3-9 和表 3-10	数值型或字符型等	逻辑值 0 或 1 （整型）	从右至左
与	&&	双目中缀				从左至右
或	||					

逻辑运算符

逻辑运算符的优先级如下：

1、！优先于双目算术运算符优先于关系运算符优先于&&优先于||。

2、单目逻辑运算符！和单目算术运算符的优先级相同，结合性均从右至左。

3、双目逻辑运算符“&&”和“||”的结合性是从左至右。

逻辑运算符

主要注意以下几点：

1、对于逻辑运算结果，“1”表示逻辑真，“0”表示逻辑假；对于逻辑运算对象“非0”表示逻辑真，“0”表示逻辑假。

2、逻辑与“&&”相当于“同时”，逻辑或“||”相当于“或者”，逻辑非“！”相当于“否定”。

3、逻辑运算符两侧的运算对象可以是任何类型的数据，如整型、实型、字符型等。

4、一旦“&&”和“||”能够确定表达式的值，则不再继续运算，即并不是所有的表达式都被运算。

逻辑表达式

逻辑表达式是用逻辑运算符将一个或多个表达式连接起来，进行逻辑运算的式子。

逻辑表达式的构成规则：单目逻辑运算符 表达式

表达式 双目逻辑运算符 表达式

其中，表达式主要是关系表达式，也可以是字符型数据或算术表达式、条件表达式、赋值表达式、逗号表达式等。由于条件、赋值、逗号运算符的优先级低于逻辑运算符，所以应注意加圆括号。

基本赋值运算符

基本赋值运算符即赋值运算符“=”，它是双目运算符，赋值运算符的左边必须是变量，右边是表达式，其作用是将一个表达式的值赋给一个变量。

名称	运算符	运算对象个数	运算规则	运算对象类型	结果类型	结合性
赋值	=	双目中缀	计算右边表达式的值赋予左边的变量	任何类型	变量的类型	从右至左

赋值运算符的优先级：

1、算术运算符优先于关系运算符优先于双目逻辑运算符优先于赋值运算符；

2、赋值运算符的结合性是从右至左。

复合赋值运算符

复合赋值运算符是在赋值运算符的前面再加一个双目运算符构成的，它是双目运算符，复合赋值运算符的左边必须是变量，右边是表达式。

复合赋值运算符的一般使用形式：

变量 双目运算符=表达式　等价于　变量=变量 双目运算符　（表达式）

复合赋值运算符的优先级：

1、算术运算符优先于关系运算符优先于双目逻辑运算符优先于复合赋值运算符。

2、复合赋值运算符和赋值运算符的优先级相同，结合性都是从右至左。

表 3-12　复合赋值运算符的说明

名称	运算符	运算对象个数	运算规则	运算对象类型	结果类型	结合性
加赋值	+=	双目 中缀	x+=y 等价于 x=x+(y)	数值型	数值型	从右至左
减赋值	-=		x-=y 等价于 x=x-(y)			
乘赋值	*=		x*=y 等价于 x=x*(y)			
除赋值	/=		x/=y 等价于 x=x/(y)			
模赋值	%=		x%=y 等价于 x=x%(y)	整型	整型	
位与赋值	&=		a&=b 等价于 a=a&(b)			
位或赋值	|=		a|=b 等价于 a=a|(b)			
位异或赋值	^=		a^=b 等价于 a=a^ (b)			
位左移赋值	<<=		a<<=b 等价于 a=a<< (b)			
位右移赋值	>>=		a>>=b 等价于 a=a>> (b)			

a+=3 ⟺ a=a+3

x*=y+8 ⟺ x=x*(y+8)

x%=3 ⟺ x=x%3

赋值表达式

由赋值运算符或复合赋值运算符将一个变量和一个表达式连接起来的表达式，称为赋值表达式。

赋值表达式的构成规则：变量=表达式

变量 复合赋值运算符 表达式

其中，表达式可以是算术表达式、关系表达式、逻辑表达式、条件表达式、赋值表达式和逗号表达式等。由于逗号运算符的优先级低于赋值运算符，所以应注意加圆括号。

```
例：    a=b=c=5              //表达式值为5，  a, b, c值为5
        a=(b=5)              //  b=5;a=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
```

逗号运算符和逗号表达式

逗号运算符又称为顺序求职运算符，它是双目运算符，运算对象是表达式，常用于for循环语句中。

名称	运算符	运算对象个数	运算规则	运算对象类型	结果类型	结合性
逗号	,	双目中缀	依次计算左边、右边表达式	任何类型表达式	右边表达式的类型	从左至右

逗号运算符和逗号表达式

- 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，级别最低
- 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: 逗号表达式使用
main()
{   int x,y=7;
    float z=4;
    x=(y=y+6,y/z);
    printf("x=%d\n",x);
}
```

运行结果：X=3

```
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
例   a=1;b=2;c=3;
     printf("%d,%d,%d",a,b,c);                  //1,2,3
     printf("%d,%d,%d",(a,b,c),b,c);            //3,2,3
```

逗号运算符和逗号表达式

逗号运算符的优先级：

1、任何运算符优先于逗号运算符，即逗号运算符的优先级最低。

2、逗号运算符的结合性是从左至右。

需要注意的是，并不是任何地方出现的逗号都是逗号运算符。很多情况下，逗号仅用作分隔符。

用逗号运算符把两个或多个表达式连接起来构成逗号表达式，构成规则：

表达式1，表达式2

其中，表达式可以是算术表达式、关系表达式、逻辑表达式、条件表达式、赋值表达式和逗号表达式等。

条件运算符和条件表达式

条件运算符是C语言中唯一的一个三目运算符，它是由？和：组合成的，其三个运算对象均为表达式，其中第一个运算对象可以是任何类型的表达式，通常理解为逻辑表达式。

条件运算符的一般使用形式：

表达式1？表达式2：表达式3

条件运算符的执行过程是先表达式1，若非0，则条件表达式的值是表达式2的值；若为0，则条件表达式的值是表达式3的值。

名称	运算符	运算对象个数	运算规则	运算对象类型	结果类型	结合性
条件	?:	三目 中缀	e1?e2:e3 若 e1 为真，取 e2 的值 若 e1 为假，取 e3 的值	表达式	e2 或 e3 的类型	从右至左

条件运算符和条件表达式

条件运算符的优先级：

1、其他运算符优先于条件运算符优先于赋值和复合赋值运算符优先于逗号运算符。

2、条件运算符的结合性是从右至左。

```
int i=1,j=1,k=2,m=0;
i==j?(k=3) : (m=-3)  // 3,k的值是3,i,j,m的值不变
```

★整型变量

❖整型数据在内存中的存放形式

- 数据在内存中以二进制补码形式存放
- 每一个整型变量在内存中占2个字节

10的原码	00	00	00	00	00	00	10	10
反码	00	00	00	00	00	00	10	10
补码	00	00	00	00	00	00	10	10
-10的原码	10	00	00	00	00	00	10	10
取绝对值	00	00	00	00	00	00	10	10
反码	11	11	11	11	11	11	01	01
补码	11	11	11	11	11	11	01	10

整数的最左二进制位是符号位，0正、1负

位逻辑运算符

位逻辑运算符将数据中的每个二进制位上的“0”或“1”作为逻辑值，按位进行逻辑运算。位逻辑运算符分为按位非、按位与、按位或和按位异或（又称按位加）四种，其中按位非是单目运算符，其余三种都是双目运算符。

名　　称	运　算　符	运算对象个数	运算规则	运算对象类型	结果类型	结　合　性
按位非	~	单目前缀	~1 为 0，~0 为 1	整型	整型	从右至左
按位与	&	双目中缀	见表 3-16			从左至右
按位或	|					
按位异或	^					

3&9=00000001　　3|9=00001011　　3^9=00001010

位逻辑运算符

1、运算对象只能是整型或字符型数据。除按位非为单目运算符外，其余均为双目运算符。

2、参与位逻辑运算时，运算对象以二进制形式进行相应的按位运算。

位逻辑运算符的优先级;

1、~优先于双目算术运算符优先于关系运算符优先于&优先于^优先于|优先于双目逻辑运算符。

2、~与单目逻辑运算符、自增、自减、单目算术运算符、长度运算符的优先级相同，结合性是从右至左。

位移位运算符

位移位运算符是将数据作为二进制数，进行向左或向右移动若干位的运算，分为左移和右移两种，均为双目运算符，其中第一个运算对象的移位对象，第二个运算对象是移动的二进制位数。

1、运算对象只能是整型或字符型数据。

2、参与位移位运算时，运算对象以二进制形式进行相应的按位运算。

表 3-17　位移位运算符的说明

名　　称	运算符	运算对象个数	运算规则	运算对象类型	结果类型	结　合　性
左移	<<	双目中缀	a<<b，a 向左移 b 位	整型	整型	从左至右
右移	>>		a>>b，a 向右移 b 位			

位移位运算符

位移位运算符的优先级：

1、算术运算符 优先于位移位运算符优先于关系运算符。

2、位移位运算符的优先级相同，结合性是从左至右。

unsigned int b=248;

b=b>>2 运算后的值是62

int x=4,y=2; 表达式(x>>y)+(x<<y)的值是 17

长度运算符

长度运算符是单目运算符，其运算对象可以是任何数据类型符或变量。

长度运算符的使用形式：sizeof（数据类型符）或sizeof（变量）

表 3-18　长度运算符的说明

名称	运算符	运算对象个数	运算规则	运算对象类型	结果类型	结合性
长度	sizeof	单目前缀	测试数据类型所占用的内存字节数	数据类型符或变量	整型	从右至左

长度运算符的优先级：

1、与单目算术运算符、单目逻辑运算符、自增和自减运算符的优先级相同。

2、上述优先级相同的运算符的结合性都是从右至左。

sizeof(int)　4　　sizeof(float)　4　　sizeof(double)　8

课堂练习

1、设int a=10；则执行语句a+=a-=a+a; 后a的值是_____。

课堂练习

1、设int a=10；则执行语句a+=a-=a+a; 后a的值是_____。

答案：-20

解析：先计算a-=a+a等价于a=a-(a+a)=10-20=-10；再计算a+=-10等价于a=a+(-10)=-10-10=-20。

课堂练习

2、设int x=10, a=0, b=25；，条件表达式x<1?a+10：b的值是（　）

A. 0

B. 1

C. 10

D. 25

课堂练习

2、设int x=10, a=0, b=25；，条件表达式x<1?a+10：b的值是（　）

A. 0

B. 1

C. 10

D. 25

答案：D

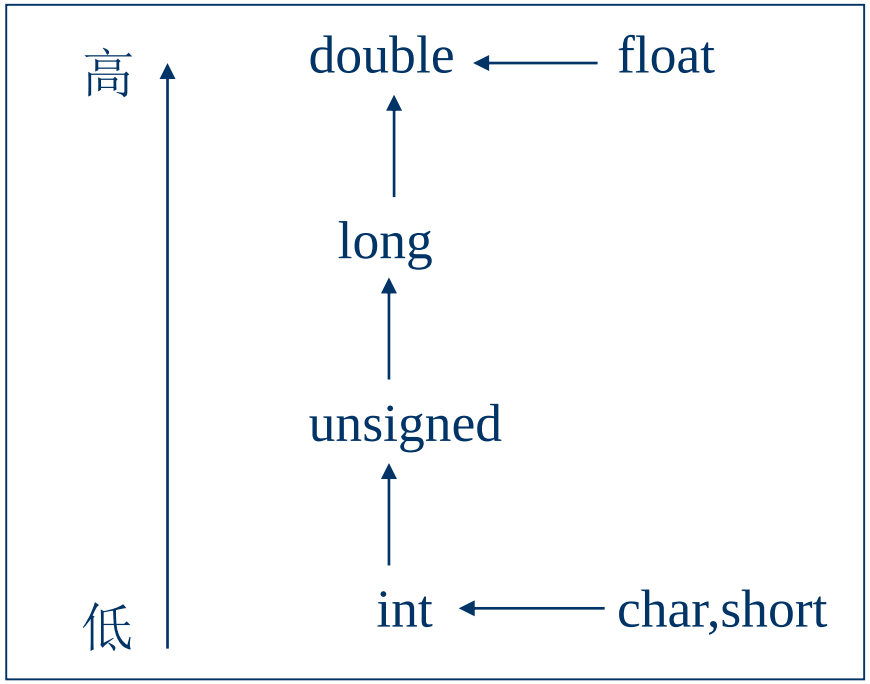
解析：设int x=10, a=0, b=25；，条件表达式x<1?a+10：b中x<1是假的取b的值为25。

第五节 数据类型转换

自动类型转换：

参与运算的各个数据都转换成数据较长的数据类型，然后计算，结果的类型就是数据长度较长的数据类型。本质上是较低类型的转换，即“就高不就低”或“就长不就短”的原则。

- **什么情况下发生**
  - **运算转换**-------不同类型数据混合运算时
  - **赋值转换**-------把一个值赋给与其类型不同的变量时
  - **输出转换**-------输出时转换成指定的输出格式
  - **函数调用转换**-------实参与形参类型不一致时转换
- **运算转换规则：不同类型数据运算时先自动转换成同一类型**



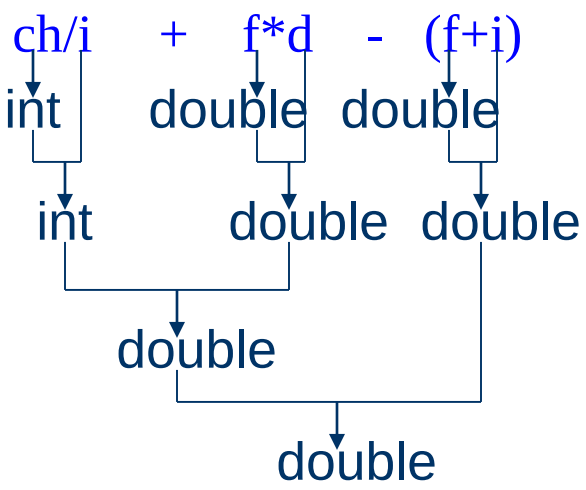
说明:

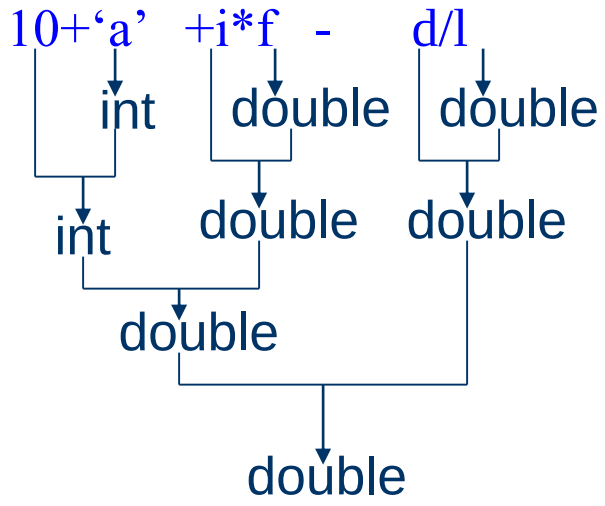
← 必定的转换

↑ 运算对象类型不同时转换

例 int i;
float f;
double d;
long l;

例 char ch;
int i;
float f;
double d;





赋值类型转换

先将运算结果的数据类型自动转换为左边变量的数据类型，然后再赋予该变量。本质上是“就左不就右”的转换原则。

例：设int a=1;float b=2.1;char c=’a’；

则表达式a=b+c的计算过程是先计算b+c，按照“就高不就低”的规则，将c转换为实型（变量c的值是97，转化为97.0），计算结果是实型，即99.1，由于变量a是整型，按照“就左不就右”的规则，所以将实型自动转换为整型赋予变量a，即99赋值给变量a。

强制类型转换

除了自动类型转换和赋值类型转换外，C语言也允许强制类型转换。

数据类型强制转换的一般格式：（数据类型符）（表达式）

C语言首先按照自动类型转换原则计算表达式的值，然后将其转换为指定的数据类型。需要注意的是，若表达式仅仅是单个常量或变量时，外面的一对圆括号可以省略；若是含有运算符的表达式，则必须用一对圆括号括起来。

int a; float b=1.9; 执行语句 a=(int)b后变量a和b的值分别是 1,1.9

★强制类型转换

强制类型转换运算符

❖一般形式：（类型名）（表达式）

例：(int)(x+y)

(int)x+y

(double)(3/2)

(int)3.6

❖说明：强制转换得到所需类型的中间变量，原变量类型不变

```
例3.8
#include <stdio>
main()
{   float  x;
    int  i;
    x=3.6;
    i=(int)x;
    printf("x=%f,i=%d",x,i);
}
结果：x=3.600000,i=3
```

表达式仅一个变量时，括号可以省略

精度损失问题

较高类型向较低类型转换时可能发生

强制类型转换

进行数据类型转换时，需要注意以下几点：

1、强制类型转换获得的是一个所需类型的中间量，原来表达式的类型并不发生变化。

2、表达式计算过程中，参与运算的常量或变量的类型自动转换是临时的，它仅产生一个临时的、类型不同的数据参与运算，常量或变量的原类型和值均不改变。

3、运算结果赋予变量的类型转换过程中，无论是自动变量的还是强制的，当数据长度较长的结果存入数据长度较短的变量时，将截去超出的部分，有可能造成错误。

4、数据类型转换将占用系统时间，过多的转换将降低程序的运行效率，所以除了必要的数据类型转换外，应尽量选择好数据类型，避免不必要的转换。

课堂练习

1、设int x=2,y=4；值为非0的表达式是（　　）

A. 1/x*y

B. 1/(x*y)

C. 1.0/x/y

D. 1/x/ (double)y

课堂练习

1、设int x=2,y=4；值为非0的表达式是（　　）

A. 1/x*y

B. 1/(x*y)

C. 1.0/x/y

D. 1/x/ (double)y

答案：C

解析：1/x*y中1、x和y都是整型，即在1整除x时，值为0，即1/x*y最终的值为0；1/(x*y)中1是整型常量，即在1整除x*y时，值为0；1.0/x/y中1.0表示1.0为实型，即在1.0整除x时，是可以保留小数的，即1.0/x/y最终的值为非0；1/x/ (double)y中1和x都是整型，即在1整除x时，值为0。

课堂练习

2、设int a=1；float b=2.1；char c= ‘a’ ；，则表达式a=b+c最后的类型为（　　）

A. int

B. float

C. double

D. char

课堂练习

2、设int a=1；float b=2.1；char c= ‘a’ ；，则表达式a=b+c最后的类型为（　　）

A. int

B. float

C. double

D. char

答案：A

解析：表达式a=b+c的计算过程是先计算b+c，按照“就高不就低”的规则，将c转换为实型，c的值为97.0，计算结果为实型，即99.1，由于变量a是整型，按照“就左不就右”的规则，则表达式a=b+c最后的类型为int型。

祝大家顺利通过考试！