如何编程序？

1、根据问题，先想出思路，不要着急写代码！
2、用 C 语言描述思路——结构化程序设计方法：先粗后细，先抽象后具体！





3、结构化程序由若干模块组成，注意一些原则：模块功能单一、低耦合...
4、攻略：从简单程序开始、从模仿开始、编程是练会的！

例：求两个数的和

```
/*      example1.1    calculate the sum of a and b*/
#include <stdio.h>
int add(int x,int y);
int main(void)
{     int a,b,sum;
      a=10;
      b=24;
      sum=add(a,b);
      printf("sum= %d\n",sum);
      return 0;
}
/* This function calculates the sum of x and y   */
int add(int x,int y)
{   int  z;
    z=x+y;
    return(z);
}
```

预处理命令

函数

注释

语句

运行结果：
sum=34

例：求两个正整数的最大公约数

欧几里德算法

输入：正整数 *m* 和正整数 *n*

输出：*m* 和 *n* 的最大公约数 (设 m>n)

步骤:

Step 1. *m* 除以 *m*，记余数为 *r*

Step 2.

如果 *r* 不等于 *0* ，

将 *n* 的值赋给 *m*, *r* 的值赋给 *n*，返回 **Step 1;**

否则，最大公约数是 *n*，输出 *n*，算法结束。

例： M=35, n=25,

输出：最大公约数是： 5

```c
#include <stdio.h>
int ComFactor(int m, int n)
{
    int r = m % n;
    while (r != 0)
    {
        m = n;  n = r;
        r = m % n;
    }
    return n;
}
int main( )
{
    int x = ComFactor(35, 25);
    printf("最大公约数是: %d\n", x);
    return 0;
}
```

问题1：

中国大学MOOC

是不是 无论我们在程序里写什么“单词”，计算机都能明白？

编程语言定义了一些有特定含义的“关键字”，计算机“只能明白”这些“词”的含义。

问题2：

中国大学MOOC

是不是 无论我们在程序里写什么“数”和“计算符号”，计算机都能明白？

计算机只能“看懂”某些类型的数据，这些“数据的类型”和相应的“操作符号”也是定义好的。

问题3：

中国大学MOOC

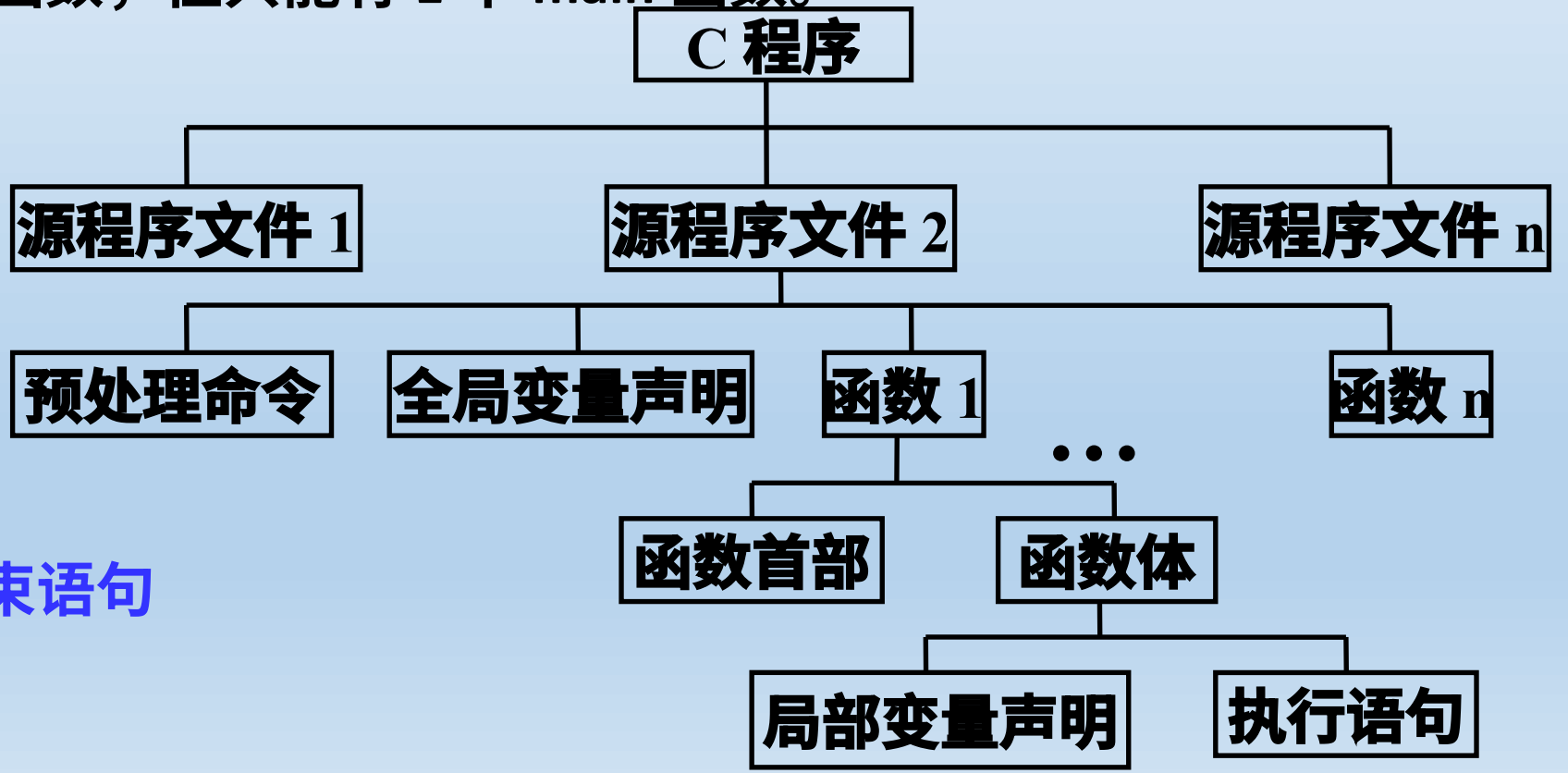
世界上可能要用“程序来表达的逻辑”纷繁复杂，程序设计语言里面得有多少种“句式”才能够用？

只需要三种“句式”：

顺序、分支、循环

C 程序结构

- 源程序、目标程序和可执行程序
- 工作空间或解决方案：
  - 一个工作空间只能生成一个可执行程序，只能有一个 main 函数
  - 一个工作空间可以有 1 个或多个源程序文件 (*.cpp 或 *.c) ， 1 个或多个头文件 (*.h) ，包括 1 个或多个函数，但只能有 1 个 main 函数。



- C 程序的结构特点
  - 大小写敏感
  - 不使用行号，以分号结束语句
  - 可使用空行和空格
  - 常用锯齿形书写格式

C 语言的词法

- **字符集**

大小写英文字母、数字、下划线以及 + - * / = , ; ? ! ” % . & ^ # ’ < > | ( ){ } [ ] ~ \ 等

- **标识符**
  - •定义：用来标识变量、常量、函数等的字符序列
  - 组成：
    - 只能由字母、数字、下划线组成，且第一个字母必须是字母或下划线
    - 大小写敏感
    - 不能使用关键字
  - 命名原则：
    - 见名知意， 常用取简、专用取繁
    - 注意混淆： l 与 I, o 与 0

- 合法标识符： a, b, c, abc, a1, b2, c3, _price, name, pi, PI
- 不合法标识符： 1a ， int ， a#3 ， b&a

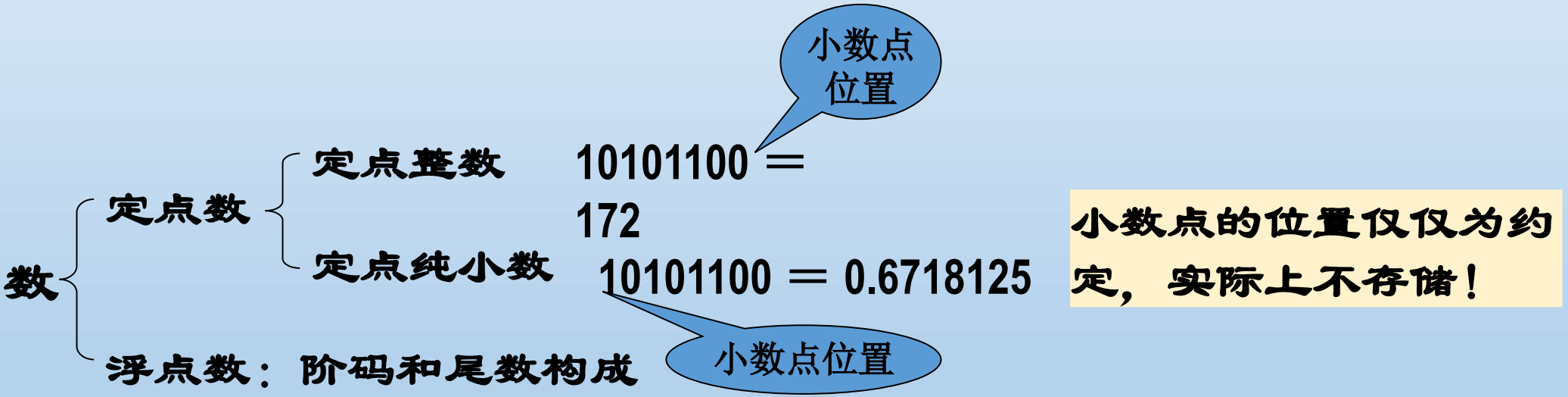
C 的关键字

（由系统定义，不能重作其它定义）

auto	break	case	char	const
continue	default	do	double	else
enum	extern	float	for	goto
if	int	long	register	return
short	signed	sizeof	static	struct
switch	typedef	unsigned	union	void
volatile	while			

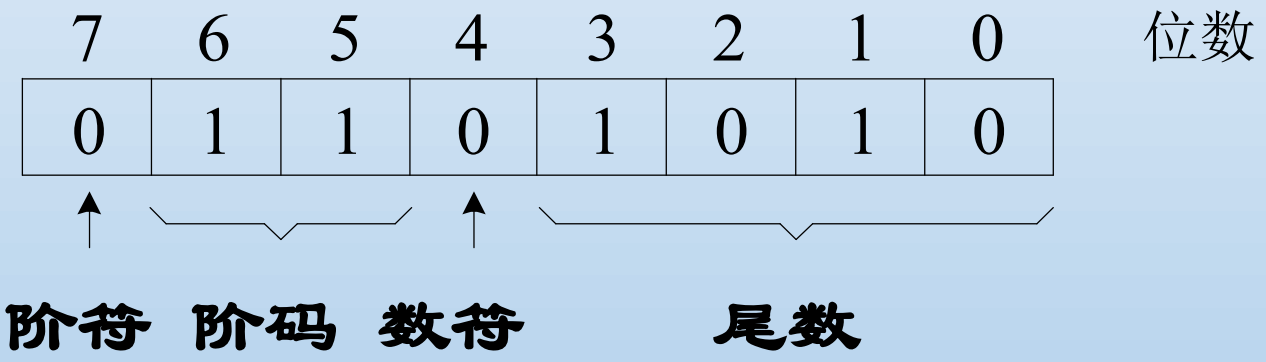
• 定点数和浮点数

- 计算机表示数的位数长度是有限的，而数的大小是无限的，差别很大。
- 为使计算机所表示的数的增大范围和提高精度，可采用浮点数。
- 定点数为小数点位置固定的数；
- 浮点数为小数点位置不固定的数；



浮点数

- **浮点数**（Floating Point Numbers），**例如：**



$$2^{(011)_2} \times (0.1010)_2$$

C 语言中的数据类型

- 整型类，表示整数；
  - 常用类型： int 4 字节（ 32 位编译器），有符号整数（补码）
  - 其它类型： short ， long ， unsigned ， unsigned short ， unsigned long
  - 例： unsigned short int 范围： 0-65535
- 实数类，表示小数；
  - float： 单精度小数， 4 字节， 6-7 个有效数字
  - double： 双精度小数， 8 字节， 15-16 个有效数字
- 字符类，表示单个字符
  - char： ASCII 编码， 1 字节，

  例：大写字母 ‘A’ 编码为 65 ，

  对应二进制为： 0100 0001 （计算机内存储表示）， 16 进制为 0x41 ；

C 语言中的常量

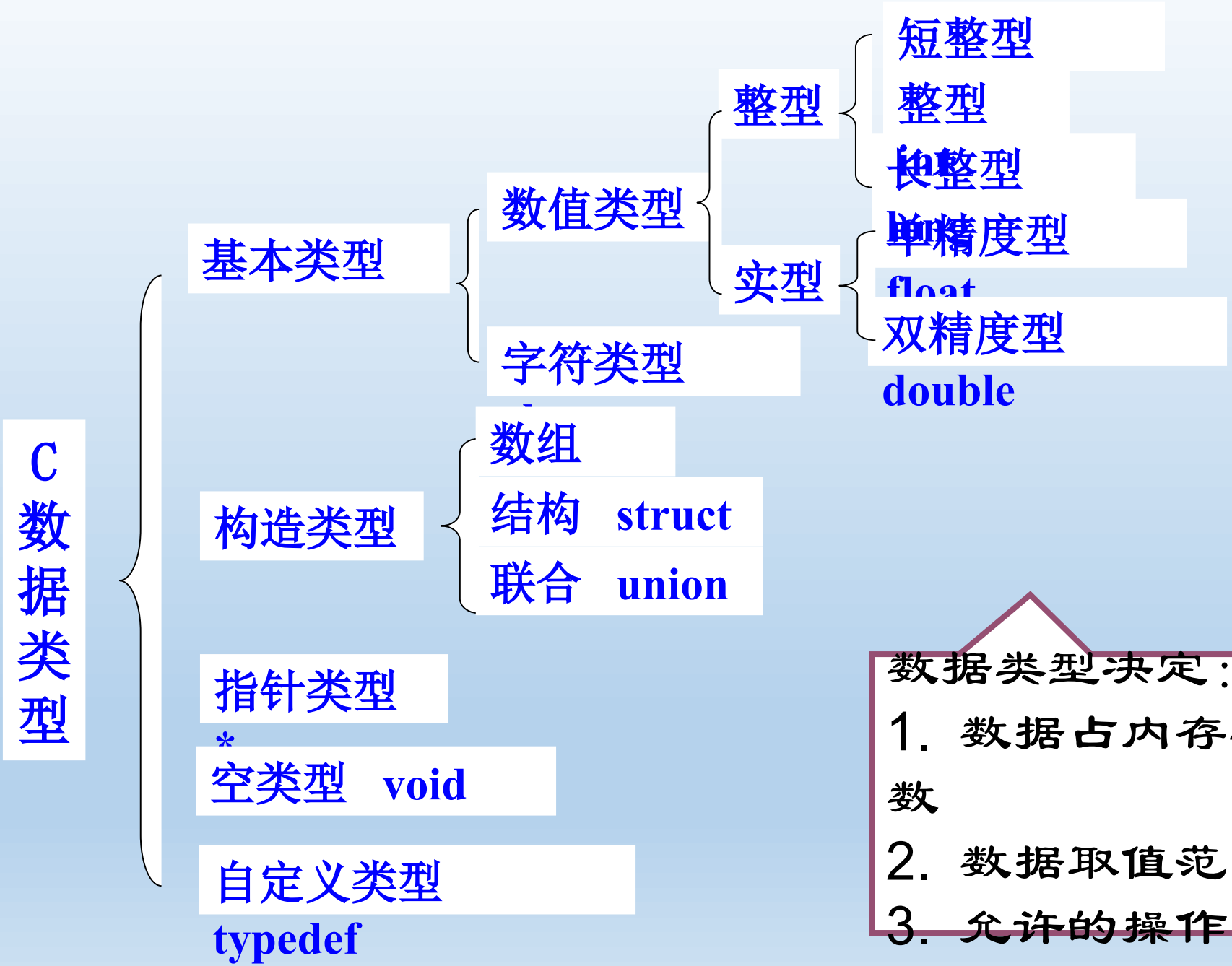
- 常量：在程序中值不改变的量，直接常量和符号常量
- 直接常量，如：
  - 整型： 45 ， 0X41 ；无符号整型： 45u ，长整型： 86L;
  - 双精度： 1.23 ， 1.23e2 ， .23e3 ；单精度： 1.23f
  - 字符： ‘A’ ， ‘a’ ， ‘1’ ， ‘\0’ ， ‘\n’ ， ‘\x41’,’\101’; （P52）
  - 字符串常量：” How are you?”, ” 你好！”
- 符号常量：用一个符号来表示的，在程序运行时不可改变的量
  - 宏： #define PI 3.14159
  - const 常量： const double PI = 3.14159; const int R = 5;
  - 标识符：用于标识程序中的常量、变量、函数名等;

C 语言中的变量

- 变量：在程序运行中值可以被修改的量，必须用一个合法的标识符来表示；
  - ✓变量必须先定义后使用，一般在函数开头部分定义（标准 C 和 C++ 的区别）
- 整型变量：表示一个整数，运行时变量的值可被修改，如：
  - int a, b, c； a = 1； a = 2; b = 3; c = a + b; // 变量定义、赋值、计算语句
- 实型变量：表示一个实数，运行时变量的值可被修改，如：
  - float f1, f2,f3 = 0; f1 = 1.2; f2 = 2.4f; f3 = f1+f2;
  - double d1, d2, d3; d1 =2.4; d2 = 3.5 – d1; d3 = d1 *d2;
- 字符型变量：表示一个字符，运行时变量的值可被修改，如：
  - char c1, c2,c3; c1 = ‘a’; c2 = c1 – 32; c3 = ‘B’ +32; c1 = ‘\x41’;
- 指针型变量：表示一个变量的地址， 32 位编译器中占 4 字节，如：
  - int a； int *pa = &a；
- 组合型变量：表示组合数据，如：
  - int a[5]; 表示 5 个整数，分别是 a[0],a[1],a[2],a[3],a[4];
  - struct STU stu1; 需要先定义组合规则：

  可以使用 stu1.name ， stu1.ID ， stu1.score 等变量；

```
struct STU{
    char name[20];
    int ID;
    double
score[5];
};
```


C数据类型
基本类型
数值类型
整型
短整型
整型
长整型
实型
单精度型
float
双精度型
double
字符类型
构造类型
数组
结构 struct
联合 union
指针类型
*
空类型 void
自定义类型
typedef
数据类型决定：
1. 数据占内存字节数
2. 数据取值范围
3. 允许的操作

例

```
#include <stdio.h>
#define   PRICE    12.5
int main(void)
{
    int   num=3;
    float   total;
    char   ch1, ch2='D';

    total=num*PRICE;
    ch1=ch2-'A'+'a';
    printf("total=%f, ch1=%c\n",total,ch1);
    return 0;
}
```

常量宏定义

变量定义

输出结果

运行结果：

```
total=37.500000, ch1=d
```

C 运算符

算术运算符：+ - * / % ++ --

关系运算符：< <= == > >= !=

逻辑运算符：! && ||

位运算符 ：<< >> ~ | ^ &

赋值运算符：= 及其扩展

条件运算符：?:

逗号运算符：,

指针运算符：* &

求字节数 ： sizeof

强制类型转换：（类型）

分量运算符：. ->

下标运算符：[]

其它 ：( ) -

学习运算符应注意：

（1）掌握运算符的功能；
（2）搞清运算符与运算量间的关系：
- 要求运算量个数
- 要求运算量类型

（3）清楚运算的优先级；

（4）明确结合方向（在一个运算量的两侧有两个相同优先级别的运算符，则按结合方向顺序处理）；

（5）明白结果的类型（即表达式的类型）。

算术运算符

- 基本算术运算符： + − * / %
  - 结合方向：从左向右
  - 优先级： − ----->* / % ------> + −

  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

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 (×)

算术运算符（续）

- 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

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

算术运算符（续）

- 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：　- ++ -- ------>* / % ----->+ -
  - 注意 ++ 、 -- 的副作用

例　-i++　？

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

关系运算符

关系运算符

种类： < <= == >= > !=

结合方向： 自左向右

优先级别：

< <= > >=	优先级（高）
== !=	优先级（低）

- 关系表达式的值：是逻辑值“真”或“假”，用 1 和 0 表示

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

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)*/
```

关系运算符（续）

- 关系运算注意：区分“=”与“==”

例　5>2>7>8 在 C 中是允许的，值为 0

例
```
int  i=1, j=7,a;
a=i+(j%4!=0);
```
则 a= 2

例　‘a’>0　结果为 1
‘A’>100　结果为 0

应避免对实数作相等或不等的判断

例　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

逻辑运算符

逻辑运算符

种类：　！ &&　||

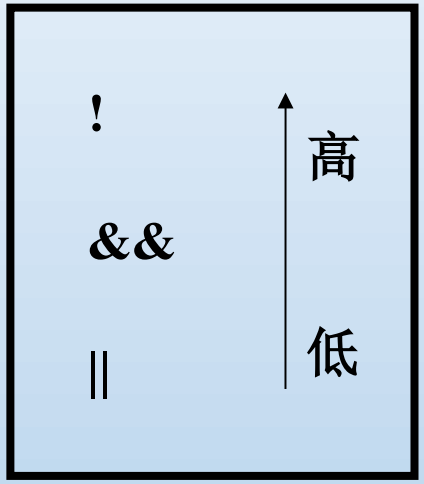
逻辑运算真值表

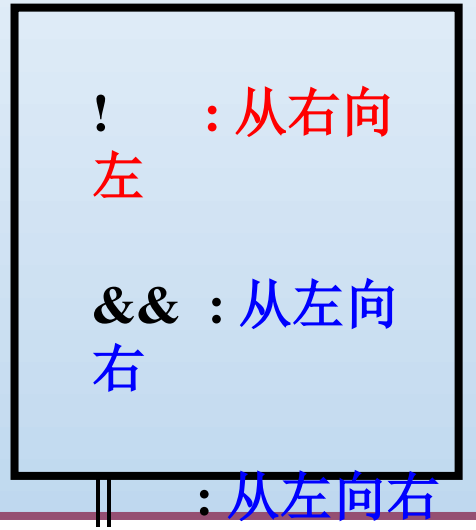
a	b	!a	!b	a&&b	a||b
真	真	假	假	真	真
真	假	假	真	假	真
假	真	真	假	假	真
假	假	真	真	假	假

- C 语言中，运算量：　0 表示“假”，
  非 0 表示“真”，
  运算结果：　0 表示“假”，
  1 表示“真”，

逻辑运算符（续）

- 优先级：
- 结合方向：

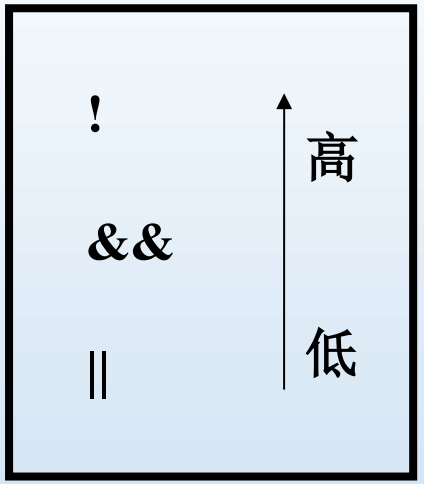




例

```
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)*/
```

- 优先级：
- 结合方向：



！ ：从右向左

&& ：从左向右

|| ：从左向右

例 a=4; b=5;

```
!a                          /* 值为 0*/
a&&b                        /* 值为 1*/
a||b                        /* 值为 1*/
!a||b                       /* 值为 1*/
4&&0||2                     /* 值为 1*/
5>3&&2||8<4-!0              /*(5>3)&&2||(8<(4-(!0)))  值为 1*/
'c'&&'d'                    /* 值为 1*/
```

逻辑运算符（续）

- **短路特性**：逻辑表达式求解时，并非所有的逻辑运算符都被执行；只有在必须执行下一个逻辑运算符才能求出表达式的解时，才执行该运算符。

```
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 */
                                /* 结果 m=0, n=1*/
例    a=1;b=2;c=3;d=4;m=1;n=1;
    (m=a>b)&&(n=c>d)
```

赋值运算符

简单赋值运算符

符号：　=

格式：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

例　a=3;
　　d=func();
　　c=d+2;

- 复合赋值运算符
  - 种类：　+=　-=　*=　/=　%=
  - 含义：　exp1 op= exp2 ⇔ exp1 = exp1 op exp2

a+=3　⟺　a=a+3

x*=y+8　⟺　x=x*(y+8)

x%=3　⟺　x=x%3

赋值运算符（续）

说明：
结合方向：自右向左
优先级：
左侧必须是变量，不能是常量或表达式

错误：
```
3=x-2*y;
a+b=3;
```

- 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

例
```
float f;
int i;
i=10;
f=i;
```
则 f=10.0

例
```
int i;
i=2.56;
```
结果 i=2;

例：
```
a=b=c=5          /* 表达式值为 5,a,b,c 值为 5*/
a=(b=10)/(c=2)   /*表达式值 5,a=5,b=10,c=2*/
```

赋值运算符（续）

- **赋值类运算符的副作用及限制**

C语言允许在一个表达式中使用一个以上赋值运算符。

※ 对人的影响和副作用，易造成误解。

1) 程序尽可能写得明了易懂一些；

2) 加上一些不必要的括号。

※ 对机器的副作用，不定解。

交换律不再适用。

例： a=12;

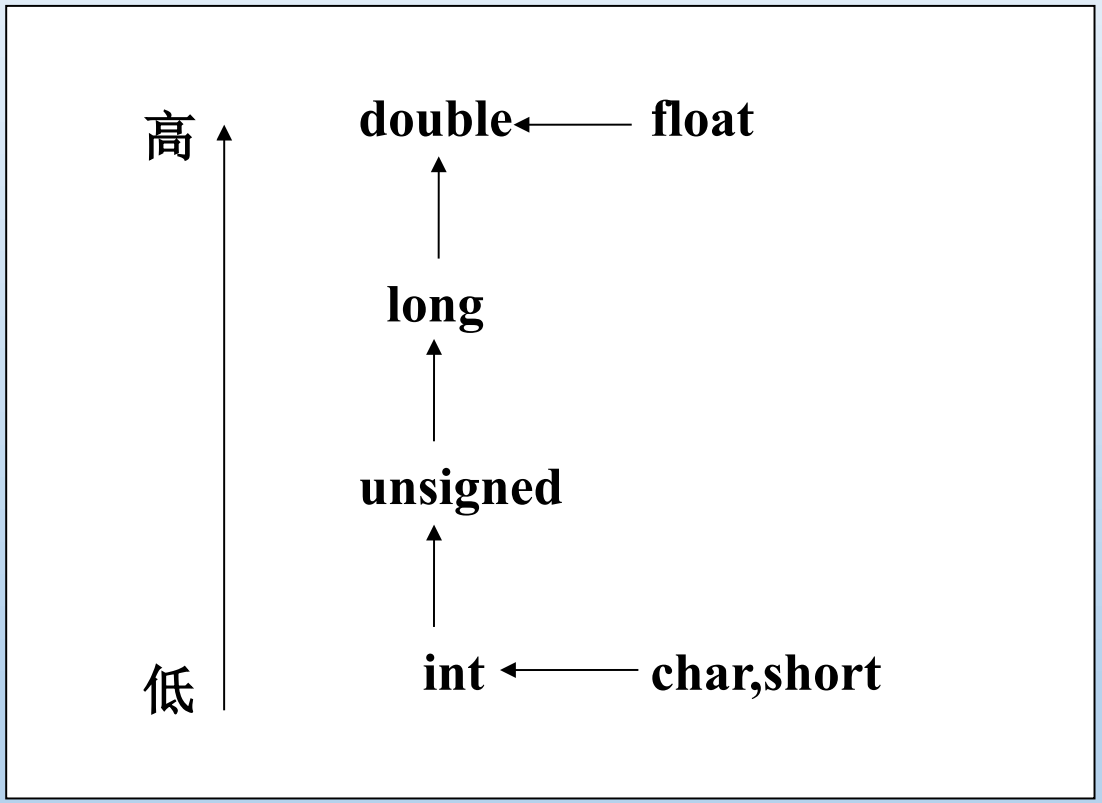
a+=a-=a*a; /*a=-264 等价于 a=a+(a=a-(a*a))*/

类型转换

- 隐式转换
  - 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
    - 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

- 显式转换（强制转换）
  - 一般形式：（类型名）表达式
  - 说明：强制转换得到所需类型，原变量类型不变

混合运算隐式转换规则



说明：

← 必定的转换

↑ 运算对象类型不同时转换

显式转换举例

例 (int)(x+y)
(int)x+y
(double)(3/2)

```
例    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
```



较高类型向较低类型转换时可能发生

逗号运算符

形式：表达式 1, 表达式 2,…… 表达式 n
结合性：从左向右
优先级： 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*/
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*/
```

计算字节运算符 sizeof

- sizeof 是一个单目的编译状态运算符，可计算出变量或类型的字节长度。

例：

```
float f;
printf(”%d”, sizeof(f));
printf(”%d”, sizeof(int));
```

Turbo C 输出结果为 4 和 2 。

- 使用 sizeof 可以增强程序的可移植性，使之不受计算机固有的数据类型长度限制。

C 语言的表达式和语句

- 表达式：由变量、常量以及运算符等连接而成的式子，表达式有确定的值；
  - 如： a + 2, 3+5
- 语句：表达式加分号构成一个完整的功能单元，是函数的基本组成；如：
  - int a, b, c ; // 变量定义
  - a = 1; // 变量赋值语句
  - b = 2; // 变量赋值语句
  - c = a+b; // 变量计算和赋值语句

C 语句概述

- C 语句：以 ; 作分隔符，编译后产生机器指令。
  - C 语句分为以下五类：
    - 控制语句：完成程序流程控制，包括条件、循环、分支等。
    - 函数调用语句：由函数调用加分号构成。
    - 表达式语句：在表达式后面加分号构成，常用的有赋值语句。
    - 空语句：仅包含一个分号的语句。
    - 复合语句：由大括号 { } 括起来的语句序列。

复合语句

格式：

```
{
    [ 数据说明部分； ]
     执行语句部分;
}
```

说明：

} 后不加分号
语法上和单一语句相同
复合语句可嵌套

```
main( )
{
   int  i , j ,  sum;
   sum=0;
   for(i=1; i<10;i++)
   {
      for(j=1;j<10;j++)
      {
        sum=i*j ;
        printf(“%d ”,sum);
      }
   }
}
```

C 语言中的函数

- 函数：由多条语句构成的，能够完成一定功能的代码组合，包括函数名，输入参数，返回值，语句等；
- C 程序由一个或多个函数构成，有且只有 1 个 main 函数，程序从 main 函数的第一条可执行语句开始执行，并在 main 函数结束；
- 标准库函数

  int printf(char *format...); int scanf(char *format[,argument,...]);
- 自定义函数

  int sum(int a, int b){int sum; sum = a + b; return sum;}

  int max(int a, int b){return a>b ? a: b;}

C 语言的输出函数

- 字符输出函数： int putchar(int ch) ，
  - 输入参数：字符的 ASCII 码，返回值：转换成 int 的 ASCII 码
  - 功能：向显示器输出一个字符；
  - 例如： char c1 = 'A'; …., putchar(c1); putchar(65);
- 格式化输出函数： int printf(char *format[,argument,...]);
  - 输入参数：指定格式和输出数据；返回值：字符数或负数
  - 功能：向显示器按指定格式输出数据
  - 常用格式： %d 整数 ; %f 单精度数 ; %c 字符 ; 转义字符' \t','\n'
  - 例如： int a = 3; double d = 5.2; float f = 3.9; char ch = 97; /* 定义变量并赋值 */

    printf("%5d\t%6.2f\nd = %f\tch = %c",a,f,d,c); /* 从输出数据 */

d,i	十进制整数	int a=567; printf (“%d”,a);	567
x,X	十六进制无符号整数	int a=255; printf(“%x”,a);	ff
o	八进制无符号整数	int a=65; printf(“%o”,a);	101
u	不带符号十进制整数	int a=567; printf(“%u”,a);	567
c	单一字符	char a=65; printf(“%c”,a);	A
s	字符串	printf(“%s”,“ABC”);	ABC
e,E	指数形式浮点小数	float a=567.789; printf(“%e”,a);	5.677890e+02
f	小数形式浮点小数	float a=567.789; printf(“%f”,a);	567.789000
	e和f中较短一	float a=567.789; printf(“%g”,a);	567.789
	“%	);	%

```
int main( )
{
        unsigned int
x=0XFFFFFFFF;
        printf("%u\n",x);
        printf("%d\n",x);
        return 0;
}
```

输出结果：

4294967295

外均应

个数应相

类型不一

例
```
int a=3, b=4;
printf(“%d %d\n”,a,b);
printf(“a=%d , b=%d\n”,a,b);
```

输出结果： 3 4

a=3, b=4

C 语言的输入函数

- 字符输入函数： int getchar(void) ，
  - 输入参数：无；返回值：字符的 ASCII 码值
  - 功能：从输入流（键盘）中获取一个字符； int ch = getchar();
  - 例如： char c1 ; …., c1 = getchar(); 可从键盘获取 1 个字符并赋给变量 c1 ；
- 格式化输入函数： int scanf(char *format[,argument,...]);
  - 输入参数：指定格式和输入数据存放地址；返回值：正确输入数据的个数；
  - 功能：从输入流（键盘）按指定格式获取数据
  - 常用格式： %d 整数 ; %f 单精度数 ; %lf 双精度数 ; %c 字符 ;
  - 例如： int a, num; double d; float f; char ch; /* 定义变量 */

    num = scanf(“%d%f%lf%c”,&a,&f,&d,&c); /* 从键盘获取数据 */
  - VS 的安全输入函数： scanf_s(“%c”, &a, sizeof(a)); /* 增加了 1 个参数 */

- 输入分隔符的指定
  - 一般以空格、 TAB 或回车键作为分隔符
  - 也可用其它字符做分隔符，但尽量不用

例 scanf(“%d%o%x”,&a,&b,&c);
printf(“a=%d,b=%d,c=%d\n”,a,b,c);

例 scanf(“%d:%d:%d”,&h,&m,&s);
输入 12:30:45↵

例 scanf(“%d,%d”,&a,&b)

例
scanf(“a=%d,b=%d,c=%d”,&a,&b,&c);
输入 a=12,b=24,c=36 ↵

- 说明：
  - 用“ %c” 格式符时，空格和转义字符作为有效字符输入
    - 输入数据时，遇以下情况认为该数据结束：
      - ❖遇空格、 TAB 、回车
      - ❖遇宽度结束
      - ❖遇非法输入
    - 注意输入函数留下的“垃圾”

如 scanf(“%c%c%c”,&c1,&

若输入 a bc↵

则 a⇒c1, ⇒c2, b ⇒c3

如 scanf(“%d%c%f”,&a,&b,&

若输入 1234a123o.26 ↵

则 1234 ⇒a, ‘a’ ⇒b, 123 =

例
```
int x;
char ch;
scanf(“%d”,&x);
ch=getchar( );
printf(“x=%d,ch=%d\n”,x,ch);
```
执行： 123↵

输出： x=123,ch=10

程序举例

例 ：输入三角形边长，求面积。

```
#include <math.h>
#include <stdio.h>
void main( )
{
   float a,b,c,s,area;
   scanf("%f%f%f",&a,&b,&c);
   s=1.0/2*(a+b+c);
   area=sqrt(s*(s-a)*(s-b)*(s-c));
   printf("a=%7.2f, b=%7.2f, c=%7.2f, s=%7.2f\n",a,b,c,s);
   printf("area=%7.2f\n",area);
}
```

文件包含预处理命令

变量定义

输入数据

输出数据

输入： 3,4,6 ↵

输出： a= 3.00, b= 4.00, c= 6.00 s= 6.50

area= 5.33

上机常见错误

1. 标点符号错误，中文标点符号：逗号，分号，双单引号，空格等
2. 变量运算前无确定值； int a,b,c; c= a+b;
3. 语句没有加分号；
4. 大括号不匹配；
5. 整除问题；
6. 数据输入错误：
   a) 双精度没有用 %lf
   b) 输入数据没有正确的分隔，输入格式” %d%d%d”，输入 1， 2， 3 等
   c) 输入没有正确地址，变量前应加 &
7. 注意不同的编译器版本的区别，例如

```c
int main()
{
    char a = 0, b = 0;
    scanf_s("%c %c", &a, sizeof(a), &b, sizeof(b));
    printf("%c %c", a, b);
    return 0;
}
```

《华为 C 语言编程规范》

ςΣ 13 17 19 版本下使用 σχανφ 要在最开头添加：

#define _CRT_SECURE_NO_WARNINGS

或者改用： σχανφ_σ