

# 第3章 最简单的C程序设计

## 3.1 顺序程序设计举例

## 3.2 数据的表现形式及其运算

## 3.3 C语句

## 3.4 数据的输入输出

## 3.1 顺序程序设计举例

**例3.1** 有人用温度计测量出用华氏法表示的温度(如 **64°F**)，今要求把它转换为以摄氏法表示的温度(如 **17.8°C**)。

➤ 解题思路：找到二者间的转换公式

$$c = \frac{5}{9}(f - 32)$$

**f**代表华氏温度，**c**代表摄氏温度



# 3.1 顺序程序设计举例

例**3.1** 有人用温度计测量出用华氏法表示的温度(如 **F**，今要求把它转换为以摄氏法表示的温度(如 **C**)。

➤ 算法：

输入 <b>f</b> 的值
$c = \frac{5}{9}(f - 32)$
输出 <b>c</b> 的值

N-S图



## 3.1 顺序程序设计举例

```
#include <stdio.h>
```

```
int main ( )
```

```
{
```

```
    float f,c;    定义f和c为单精度浮点型变量
```

```
    f=64.0;    指定f的值
```

```
    c=(5.0/9)*(f-32);    计算c的值
```

```
    printf("f=%f\n c=%f\n",f,c);
```

```
    return 0;    输出f和c的值
```

```
}
```

```
f=64.000000
c=17.777778
```



## 3.1 顺序程序设计举例

例**3.2** 计算存款利息。有**1000**元，想存一年。有三种方法可选：

**(1)**活期，年利率为 **$r_1$**

**(2)**一年期定期，年利率为 **$r_2$**

**(3)**存两次半年定期，年利率为 **$r_3$**

请分别计算出一年后按三种方法所得到的本息和。



# 3.1 顺序程序设计举例

➤ 解题思路：确定计算本息和的公式。

从数学知识可知：若存款额为**p0**，则：

活期存款一年后本息和为：

$$p1 = p0(1 + r1)$$

一年期定期存款，一年后本息和为：

$$p2 = p0(1 + r2)$$

两次半年定期存款，一年后本息和为：

$$p3 = p0(1 + \frac{r3}{2})(1 + \frac{r3}{2})$$



# 3.1 顺序程序设计举例

➤ 算法:

输入p0,r1,r2,r3的值
计算 $p1=p0(1+r1)$
计算 $p2=p0(1+r2)$
计算 $p3=p0(1+\frac{r3}{2})(1+\frac{r3}{2})$
输出p1,p2,p3



# 3.1 顺序程序设计举例

```
#include <stdio.h>
```

定义变量同时赋予初值

```
int main ( )
```

```
{float p0=1000, r1=0.0036, r2=0.0225,  
    r3=0.0198, p1, p2, p3;
```

```
    p1 = p0 * (1 + r1);
```

```
    p2 = p0 * (1 + r2);
```

```
    p3 = p0 * (1 + r3/2) * (1 + r3/2);
```

```
    printf("%f\n%f\n%f\n", p1, p2, p3);
```

```
    return 0;
```

```
}
```

```
1003.599976  
1022.500000  
1019.898010
```





## **3.2 数据的表现形式及其运算**

**3.2.1 常量和变量**

**3.2.2 数据类型**

**3.2.3 整型数据**

**3.2.4 字符型数据**

**3.2.5 浮点型数据**

**3.2.6 怎样确定常量的类型**

**3.2.7 运算符和表达式**



## 3.2.1 常量和变量

**1.常量：**在程序运行过程中，其值不能被改变的**量**

➤ **整型常量：** ..., -2, -1, 0, 1,2,3, ...

➤ **实型常量(浮点型)**

◆ **十进制小数形式：**如**0.34**    **-56.79**    **0.0**

◆ **指数形式：**如**12.34e3** (代表**12.34×10<sup>3</sup>**)

➤ **字符常量：**如**'?'**, **'a'**, **'1'**, 以及不能显示的符号

◆ **转义字符：**如**'\n'**

➤ **字符串常量：**如**"boy"**

➤ **符号常量：** **#define** **PI** **3.1416**

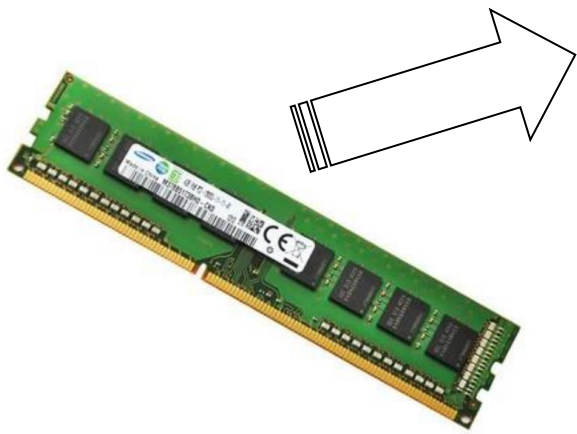


## 3.2.1 常量和变量

**2. 变量：**在程序运行期间，变量的值是可以改变的

- 变量必须**先定义，后使用**
- 定义变量时指定该变量的**名字和类型**
- **类型说明符 变量名标识符，变量名标识符，...；**
- **变量名和变量值**是两个不同的概念
- 变量名实际上是以一个名字代表的一个**存储地址**
- 从变量中取值，实际上是通过变量名找到相应的内存地址，从该存储单元中读取数据





## 3.2.1 常量和变量

3. 常变量: **const int a=3;**

4. 标识符: 一个对象的名字

大小写字母是不同的字符

- C 语言规定标识符只能由**字母**、**数字**和**下划线****3**种字符组成, 且**第一个字符必须为字母或下划线**
- 合法的标识符: 如**sum, average, \_total, Class, day, BASIC, li\_ling**
- 不合法的标识符: **M.D.John, ¥123, #33, 3D64, a>b**



## 3.2.2 数据类型

- 所谓**类型**，就是对数据分配存储单元的安排，包括存储单元的长度(占多少字节)以及数据的存储形式
- 不同的类型分配不同的长度和存储形式，
- 常用的有整型(**int**)，浮点型(**float**)，字符型(**char**)



## 3.2.2 数据类型

C语言允许使用的数据类型：

### ➤ 基本类型

#### ◆ 整型类型

- 基本整型
- 短整型
- 长整型
- 双长整型
- 字符型
- 布尔型

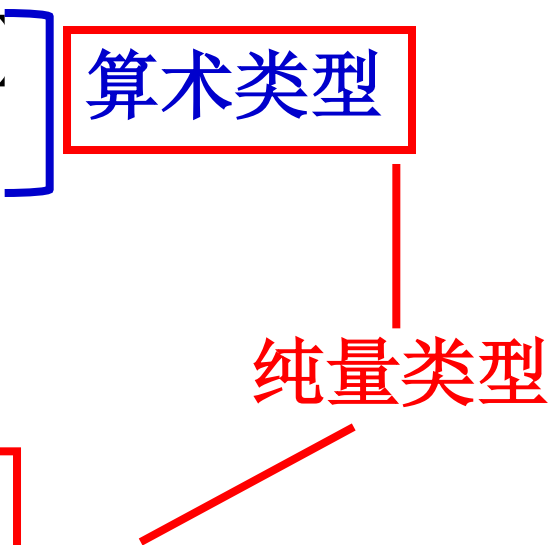
#### ◆ 浮点类型

- 单精度浮点型
- 双精度浮点型
- 复数浮点型



## 3.2.2 数据类型

C语言允许使用的数据类型：

- 基本类型(..]
  - 枚举类型
  - 空类型
  - 派生类型
    - ◆ 指针类型
    - ◆ 数组类型
    - ◆ 结构体类型
    - ◆ 共用体类型
    - ◆ 函数类型
- 算术类型
- 纯量类型
- 





## 3.2.3 整型数据

### 1. 整型数据的分类

#### ➤ 最基本的整型类型

- ◆ 基本整型(**int**型): 占**2**个或**4**个字节
- ◆ 短整型(**short int**): **VC++6.0**中占**2**个字节
- ◆ 长整型(**long int**): **VC++6.0**中占**4**个字节
- ◆ 双长整型(**long long int**): **C99**新增的



## 3.2.3 整型数据

### 1. 整型数据的分类

### 2. 整型变量的符号属性

◆整型变量的值的范围包括负数到正

◆整型变量的定义的格式为：

类型说明符 变量名标识符，变量名标识符，... ；

可以将变量定义为“无符号”类型

◆扩充的整形类型：



## 3.2.3 整型数据

扩充的整型类型:

- 有符号基本整型 **[signed] int;**
- 无符号基本整型 **unsigned int;**
- 有符号短整型 **[signed] short [int];**
- 无符号短整型 **unsigned short [int];**
- 有符号长整型 **[signed] long [int];**
- 无符号长整型 **unsigned long [int]**
- 有符号双长整型 **[signed] long long [int];**
- 无符号双长整型 **unsigned long long [int]** 

## ➤ 整数类型的有关数据：详见P45

类型	类型说明符	长度	数的范围
基本型	int	2字节	$-32768 \sim 32767$
短整型	short	2字节	$-2^{15} \sim 2^{15}-1$
长整型	long	4字节	$-2^{31} \sim 2^{31}-1$
无符号整型	unsigned	2字节	$0 \sim 65535$
无符号短整型	unsigned short	2字节	$0 \sim 65535$
无符号长整型	unsigned long	4字节	$0 \sim (2^{32}-1)$

## 3.2.4 字符型数据

- 字符是按其代码(整数)形式存储的
- **C99**把字符型数据作为整数类型的一种
- 字符型数据在使用上有自己的特点
  - ◆ 字符型变量的定义的格式为：  
类型说明符 变量名标识符, 变量名标识符, ... ;



## 3.2.4 字符型数据

### 1. 字符与字符代码

大多数系统采用**ASCII**字符集

◆字母：**A ~Z, a ~z**

◆数字：**0~9**

◆专门符号：**29个：! " # & ' ( ) \*等**

◆空格符：空格、水平制表符、换行等

◆不能显示的字符：空(**null**)字符(以'**\0**'表示)、警告(以'**\a**'表示)、退格(以'**\b**'表示)、回车(以'**\r**'表示)等



## 3.2.4 字符型数据

➤ 字符'1'和整数1是不同的概念：

◆ 字符'1'只是代表一个形状为'1'的符号，在需要时按原样输出，在内存中以**ASCII**码形式存储，占**1**个字节

0	0	1	1	0	0	0	1
---	---	---	---	---	---	---	---

◆ 整数1是以整数存储方式(二进制补码方式)存储的，占**2**个或**4**个字节

0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---



## 3.2.4 字符型数据

### 2. 字符变量

➤ 用类型符**char**定义字符变量

◆ **char c = '?';**

系统把“?”的**ASCII**代码**63**赋给变量**c**

◆ **printf("%d %c\n",c,c);**

◆ 输出结果是:

**63 ?**





## 3.2.5 浮点型数据

浮点型数据是用来表示具有小数点的实数

### ➤ **float**型(单精度浮点型)

- ◆编译系统为**float**型变量分配**4**个字节
- ◆数值以规范化的二进制数指数形式存放

参见主教材图**3.12**



## 3.2.5 浮点型数据

浮点型数据是用来表示具有小数点的实数

### ➤ **float**型(单精度浮点型)

- ◆编译系统为**float**型变量分配**4**个字节

- ◆**6**位有效数字

### ➤ **double**型(双精度浮点型)

- ◆编译系统为**double**型变量分配**8**个字节

- ◆**15**位有效数字

### ➤ **long double**(长双精度)型



## 3.2.6 怎样确定常量的类型

- 字符常量：由单撇号括起来的单个字符或转义字符
- 整型常量：不带小数点的数值
  - ◆ 系统根据数值的大小确定**int**型还是**long**型等
- 浮点型常量：凡以小数形式或指数形式出现的实数
  - ◆ **C**编译系统把浮点型常量都按双精度处理
  - ◆ 分配**8**个字节



## 3.2.7 运算符和表达式

### 1. 基本的算术运算符:

**+** : 正号运算符(单目运算符)

**-** : 负号运算符(单目运算符)

**\*** : 乘法运算符

**/** : 除法运算符

**%** : 求余运算符

**+** : 加法运算符

**-** : 减法运算符



## 3.2.7 运算符和表达式

### 说明

- 两个整数相除的结果为整数
  - ◆ 如**5/3**的结果值为1，舍去小数部分,**4/5=?**
  - ◆ 如果除数或被除数中有一个为负值，舍入方向不固定。例如，**-5/3**，有的系统中得到的结果为**-1**，在有的系统中则得到结果为**-2**
  - ◆ **VC++**采取“向零取整”的方法  
如**5/3=1**，**-5/3=-1**，取整后向零靠拢
- **%** 运算符要求参加运算的运算对象(即操作数)为整数，结果也是整数。如**8%3**，结果为**2**



## 3.2.7 运算符和表达式

### 2. 自增、自减运算符:

➤作用是使变量的值 **1** 或减 **1**

◆**++i, --i**: 在使用**i**之前, 先使**i**的值加 (减) **1**

◆**i++, i--**: 在使用**i**之后, 使**i**的值加 (减) **1**



令i的值等于3，请分析下列语句：

①  $j = ++i$ ;

i的值先变成4，再赋给j，j和i的值均为4

②  $j = i++$ ;

先将i的值3赋给j，j的值为3，然后i变为4



注意：

(1) 自增运算符（++），自减运算符（--），只能用于变量，而不能用于常量或表达式，

(2) ++和--的结合方向是“自右至左”。

## 3.2.7 运算符和表达式

### 3. 算术表达式和运算符的优先级与结合性：

- 用算术运算符和括号将运算对象（也称操作数）连接起来的、符合 C 语法规则的式子，称为 C 算术表达式
- 运算对象包括常量、变量、函数等
- C 语言规定了运算符的优先级和结合性





## 3.2.7 运算符和表达式

### 4. 不同类型数据间的混合运算:

- (1) **+**、**-**、**\***、**/** 运算的两个数中有一个数为**float**或**double**型，结果是**double**型。系统将**float**型数据都先转换为**double**型，然后进行运算
- (2) 如果**int**型与**float**或**double**型数据进行运算，先把**int**型和**float**型数据转换为**double**型，然后进行运算，结果是**double**型
- (3) 字符型数据与整型数据进行运算，就是把字符的**ASCII**代码与整型数据进行运算



## 3.2.7 运算符和表达式

例**3.3** 给定一个大写字母，要求用小写字母输出。

➤ 解题思路：

- ◆ 关键是找到大、小写字母间的内在联系
- ◆ 同一个字母，用小写表示的字符的**ASCII**代码比用大写表示的字符的**ASCII**代码大**32**



## 3.2.7 运算符和表达式

```
#include <stdio.h>
```

```
int main ( )
```

```
{
```

```
    char c1,c2;
```

```
    c1='A'; 将字符'A'的ASCII代码65放到c1中
```

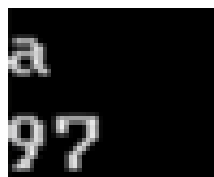
```
    c2=c1+32; 将65+32的结果放到c2中
```

```
    printf("%c\n",c2); 用字符形式输出
```

```
    printf("%d\n",c2); 用十进制形式输出
```

```
    return 0;
```

```
}
```



```
a
97
```



## 3.2.7 运算符和表达式

### 5. 强制类型转换运算符

➤ 强制类型转换运算符的一般形式为

(类型名) (表达式)

◆ **(double)a** (将 a 转换成**double**类型)

◆ **(int) (x+y)** (将**x+y**的值转换成**int**型)

◆ **(float)(5%3)** (将**5%3**的值转换成**float**型)

➤ 有两种类型转换

◆ 系统自动进行的类型转换

◆ 强制类型转换



## 3.2.7 运算符和表达式

### 6. C 运算符

- |                  |                   |
|------------------|-------------------|
| <b>(1)</b> 算术运算符 | (+ - * / % ++ --) |
| <b>(2)</b> 关系运算符 | (> < == >= <= !=) |
| <b>(3)</b> 逻辑运算符 | ! &&              |
| <b>(4)</b> 位运算符  | << >> ~   ^ &     |
| <b>(5)</b> 赋值运算符 | (=及其扩展赋值运算符)      |
| <b>(6)</b> 条件运算符 | (? : )            |



## 3.2.7 运算符和表达式

### 6. C 运算符

- |                |                   |
|----------------|-------------------|
| (7) 逗号运算符      | ( , )             |
| (8) 指针运算符      | ( * 和 & )         |
| (9) 求字节数运算符    | ( <b>sizeof</b> ) |
| (10) 强制类型转换运算符 | ( ( 类型 ) )        |
| (11) 成员运算符     | ( . -> )          |
| (12) 下标运算符     | ( [ ] )           |
| (13) 其他        | ( 如函数调用运算符 ( ) )  |



# C中各类运算符的优先级：

负号

初等运算符

( )、[ ]、->、•

单目运算符

!、++、--、-、(类型)

算术运算符

\*, /, %, +, -

关系运算符

<, >, <=, >=, ==, !=

逻辑运算符

&&, ||

条件运算符

?:

赋值运算符

=, +=, -=, \*=, /=, %=

逗号运算符

,

## 3.3 C语句

### 3.3.1 C语句的作用和分类

### 3.3.2 最基本的语句-----赋值语句





## 3.3.1 C语句的作用和分类

C语句分为以下**5**类：

- (1) 控制语句： **if**、**switch**、**for**、**while**、**do...while**、**continue**、**break**、**return**、**goto**等
- (2) 函数调用语句
- (3) 表达式语句
- (4) 空语句
- (5) 复合语句



## 3.3.2 最基本的语句----赋值语句

➤在C程序中，最常用的语句是：

◆赋值语句-表达式语句

◆输入输出语句

➤其中最基本的是赋值语句

◆一般形式为：

变量 赋值运算符 表达式;

如  **$i=i+1$ ;**



## 3.3.2 最基本的语句----赋值语句

例**3.4** 给出三角形的三边长，求三角形面积。



## 3.3.2 最基本的语句----赋值语句

- 解题思路： 假设给定的三个边符合构成三角形的条件
- 关键是找到求三角形面积的公式
- 公式为：

$$area = \sqrt{s(s-a)(s-b)(s-c)}$$

其中 $s=(a+b+c)/2$



```
#include <stdio.h>
```

```
#include <math.h>
```

```
int main ( )
```

```
{ double a,b,c,s,area;
```

```
  a=3.67;
```

```
  b=5.43;
```

```
  c=6.21;
```

对边长a、b、c赋值

```
  s=(a+b+c)/2;  计算s
```

计算area

```
  area=sqrt(s*(s-a)*(s-b)*(s-c));
```

```
  printf("a=%f\tb=%f\t%f\n",a,b,c);
```

```
  printf("area=%f\n",area);
```

```
  return 0;
```

```
}
```



```
#include <stdio.h>
```

```
#include <math.h> 调用数学函数加此行
```

```
int main ( )
```

```
{ double a,b,c,s,area;
```

```
  a=3.67;
```

```
  b=5.43;
```

```
  c=6.21;
```

```
  s=(a+b+c)/2;
```

```
  area=sqrt(s*(s-a)*(s-b)*(s-c));
```

```
  printf("a=%f\tb=%f\t%f\n",a,b,c);
```

```
  printf("area=%f\n",area);
```

```
  return 0;
```

```
}
```

数学函数，计算平方根



```
#include <stdio.h>
```

```
#include <math.h> 调用数学函数加此行
```

```
int main ( )
```

```
{ double a,b,c,s,area;
```

```
  a=3.67;
```

```
  b=5.43;
```

```
  c=6.21;
```

```
  s=(a+b+c)/2;
```

```
  area=sqrt(s*(s-a)*(s-b)*(s-c));
```

```
  printf("a=%f\tb=%f\t%f\n",a,b,c);
```

```
  printf("area=%f\n",area);
```

转义字符，使输出位置跳到下一个tab位置

```
a=3.670000
```

```
b=5.430000
```

```
6.210000
```

```
area=9.903431
```



## ➤ 归纳总结:

### 1. 赋值运算符

- ◆ “=” 是赋值运算符
- ◆ 作用是将一个数据赋给一个变量
- ◆ 也可以将一个表达式的值赋给一个变量





## ➤ 归纳总结:

### 1. 赋值运算符

### 2. 复合的赋值运算符

◆ 在赋值符“=”之前加上其他运算符，可以构成复合的运算符

◆  $a += 3$                       等价于    $a = a + 3$



## ➤ 归纳总结:

### 1. 赋值运算符

### 2. 复合的赋值运算符

### 3. 赋值表达式

#### ◆ 一般形式为:

变量 赋值运算符 表达式

#### ◆ 对赋值表达式求解的过程:

- 求赋值运算符**右侧**的“表达式”的值
- 赋给赋值运算符**左侧**的变量



## ➤ 归纳总结:

### 1. 赋值运算符

### 2. 复合的赋值运算符

### 3. 赋值表达式

◆ 赋值表达式 “ **$a=3*5$** ” 的值为**15**，对表达式求解后，变量 **$a$** 的值和表达式的值都是**15**

◆ “ **$a=(b=5)$** ” 和 “ **$a=b=5$** ” 等价

◆ “ **$a=b$** ” 和 “ **$b=a$** ” 含义不同



## ➤ 归纳总结:

**1. 赋值运算符**

**2. 复合的赋值运算符**

**3. 赋值表达式**

**4. 赋值过程中的类型转换**

◆ 两侧类型一致时，直接赋值

◆ 两侧类型不一致，但都是算术类型时，自动将右侧的类型转换为左侧类型后赋值

◆ 定义变量时要防止数据溢出



## ➤ 归纳总结:

**1.赋值运算符**

**2.复合的赋值运算符**

**3.赋值表达式**

**4.赋值过程中的类型转换**

**5.赋值表达式和赋值语句**

- ◆ 赋值表达式的末尾没有分号，而赋值语句有分号
- ◆ 一个表达式可以包含赋值表达式，但决不能包含赋值语句



## ➤ 归纳总结:

1. 赋值运算符
2. 复合的赋值运算符
3. 赋值表达式
4. 赋值过程中的类型转换
5. 赋值表达式和赋值语句
6. 变量赋初值

**int a=3,b=3,c;**

**int a=3;** 相当于 **int a; a=3;**



# 练习题1

- ①让**x**等于 【**a**与**b**乘**c**的和，除以**d**的余数】；
- ②让**x**等于 【**a**与**b**的商，加**c**与**d**的乘积，其结果乘**e**】；
- ③使用 ‘\*=’ 符号，让**a**乘以**b**与**c**的和，这一结果赋给**a**；
- ④让**x**的结果 在**a**，**b**同真，或**c**，**d**同真时，为真，其余情况为假；
- ⑤万有引力公式：令 $\mathbf{F} = \frac{GMm}{R^2}$ （其中**G**,**M**,**m**,**R**都为变量）；
- ⑥倍角公式：令 $\mathbf{x} = \sin(2\mathbf{a})$ ,  $\mathbf{y} = 2\sin(\mathbf{a})\cos(\mathbf{a})$ （**a**为变量）；
- ⑦半角公式：令 $\mathbf{x} = \sqrt{\frac{1 - \cos \mathbf{a}}{1 + \cos \mathbf{a}}}$ （**a**为变量）；

## 练习题2

⑧让**x**为斐波那契数列第**n**个的值（**n**为变量）：

$$x = \frac{1}{\sqrt{5}} \left[ \left( \frac{1+\sqrt{5}}{2} \right)^n - \left( \frac{1-\sqrt{5}}{2} \right)^n \right]$$

提示：**x<sup>y</sup>** 可以用**pow（x，y）**函数实现

⑨**relu**函数：使用三元运算符?:使得下式成立（**x**为变量）：

$$y = \begin{cases} 0 & , x < 0 \\ x & , x \geq 0 \end{cases}$$

⑩对数换底公式：

令**x** =  $\ln(a)$ ，**y** =  $\frac{\log_{10}a}{\log_{10}e}$ （**a**为变量，**e**为自然常数,用**2.718**代替）

提示：**ln()** 在**c**语言中为**log()**， $\log_{10}$ 在**c**语言中为**log10（）**