

内容回顾

- **C历史：C语言之父、C与Unix的关系等**
- **简单的C程序：**
 - ◆ **main函数**
 - ◆ **函数的形式、函数的调用**
 - ◆ **printf 和 scanf**
- **C程序上机流程：编写-编译-连接-运行**

第2章 最简单的C程序设计

2.1 顺序程序设计举例

2.2 数据的类型及存储形式

2.3 用表达式进行数据运算

2.3 C语句

2.4 数据的输入输出

2.1 顺序程序设计举例

例 输入三角形的三边长，求三角形面积。



2.1 顺序程序设计举例

- 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 $s = (a+b+c)/2$



C程序设计

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

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

```
int main ( )
```

```
{ float a,b,c,s,area; 新的数据类型 %f
```

```
scanf("%f,%f,%f",&a,&b,&c);
```

```
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;
```

```
}
```



C程序设计

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

```
#include <math.h> 调用数学函数加此行
```

```
int main ( )
```

```
{
```

```
float a,b,c;
```

数学函数，计算平方根

```
scanf("%f,%f,%f",&a,&b,&c);
```

```
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;
```

```
}
```



C程序设计

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

```
#include <math.h> 调用数学函数加此行
```

```
int main ( )
```

```
{
```

```
float a,b,c;
```

```
scanf("%f%f%f",&a,&b,&c);
```

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

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

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

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

```
return 0;
```

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
```



思考？

➤ 此函数有什么问题么？

◆ 未判断**a,b,c**是否能构成三角形，程序并不健壮（**Robustness**）

◆ **if(_____)**

输出这不是一个三角形

else 计算面积并输出

2.1 顺序程序设计举例

例 据估计，**2014**年阿里巴巴在美国上市之后，马云的财富达到**1400**亿美元，假设年增长率为**20%**，计算到**2030**年其财富？



2.1 顺序程序设计举例

➤ 解题思路：确定计算财产公式。

从数学知识可知：若初始财富为**p0**，**y**年后的财富为**p1**，则：

$$p1 = p0 * (1 + r)^y$$



2.1 顺序程序设计举例

➤ 算法:

输入 p_0, r, y 的值
计算 $p_1 = p_0(1+r)^y$
输出 p_1



2.1 顺序程序设计举例

```
#include <stdio.h>
#include <math.h>
int main ( )
{ double p0, p1,r;
  int y = 2030-2014;
  p0 = 1400000000000;
  r=0.2;
  p1 = p0 * pow(1 + r, y);
  printf("p1=%of\n",p1);
  return 0;
}
```



小结

➤ 新的数据类型，**float**和**double**

- ◆ **float**是单精度浮点数

- ◆ **double**是双精度浮点数

- ◆ **double**比**float**精度更高，更准确

- ◆ 读取**float**和读取**double**分别用**%f**和**%lf**

➤ 新的头文件**<math.h>**

- ◆ **sqrt, pow, sin, cos,** 等等（附录）

2.2 数据类型及存储形式

2.2.1 数据类型

2.2.2 常量和变量

2.2.3 整型数据

2.2.4 字符型数据

2.2.5 浮点型数据

2.2.6 怎样确定常量的类型

2.2.7 运算符和表达式



2.2.1 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



2.2.1 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



2.2.1 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**
-
- ```
graph TD; A[基本类型] --- B[]; B --- C[枚举类型]; B --- D[算术类型]; E[纯量类型] --- F[指针类型];
```



## 2.2.2 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：**如**1000**，**12345**，**0**，**-345**

➤ **实型常量**

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

◆ **指数形式：**如**12.34e3** (代表 **$12.34 \times 10^3$** )

➤ **字符常量：**如**'?'**

◆ **转义字符：**如**'\n'**

➤ **字符串常量：**如**"boy"**

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



## 2.2.2 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**
- **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



## 2.2.2 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**



## 小测试

➤ 以下哪个是合法的标识符？

- |               |               |                |                |
|---------------|---------------|----------------|----------------|
| <b>A. 2b</b>  | <b>B. sb</b>  | <b>C. A_</b>   | <b>D. __x_</b> |
| <b>E. a?b</b> | <b>F. h(3</b> | <b>G. else</b> | <b>H. 4.h</b>  |

## 2.2.3 整型数据

### ➤ 进制回顾

- ◆ 字节、位的概念
- ◆ 十进制、二进制、八进制、十六进制的表示
- ◆ 进制之间的互相转换
- ◆ 问题：在C程序中如何区分是何种进制的数呢
  - 0开头的数为八进制数
  - 0x开头的数为十六进制数



## 小测试

- 已知短整型数在计算机中用两个字节表示，求**123**的二进制、八进制、十六进制表示？

## 2.2.3 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





## 2.2.3 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数

- 问题：如何区分正数和负数（随后介绍）

- ◆可以将变量定义为“无符号”类型

- ◆扩充的整形类型：



## 2.2.3 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]** 

## 2.2.3 整型数据

### 整型数据的内存表示

- 无符号数只表示非负数，所有位全部表示数值
  - ◆ 已知无符号短整型数据在内存中占**2**个字节
    - 请问无符号短整型数据的表示范围？
    - **11111111 11111111**表示多少？
- 有符号数可表示负数、**0**、正数
  - ◆ 那负数该怎么表示呢？
    - 最高位写**1**，其他位写绝对值
    - 有什么问题呢？（**+0**和**-0**）
    - 补码表示法



## 2.2.3 整型数据

所有整型数据在计算机中均用补码表示

- 若正整数：其补码就是原码
- 若负整数：则先得到正数的原码，再按位取反加**1**
  - ◆ 如何根据十进制数，写出其补码？
    - **-2**的补码表示？



## 2.2.3 整型数据

如何理解计算机中的整型数[难！ 仔细理解]

➤ 内存中怎么表示？ + 如何解读这个表示

➤ 例如，**65535**这个数

◆ 在计算机中用补码表示为？ [提问]

◆ 如果是 **unsigned short int x = 65535;**

● 理解为无符号数，最高位不是符号位

◆ 如果是 **short int x = 65535;**

● 理解为有符号数，最高位是符号位



## 2.2.3 整型数据

### 整型数据的溢出现象

- 数据超出其表示范围，造成的与预期不一致的情况
- 小测试，**16**位的短整型数据**x**

```
short int x = 32767; // short int x = 65535;
```

```
x = x+1;
```

```
printf("%d", x);
```

请问打印结果？



## 2.2.4 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



## 2.2.4 字符型数据

### 1. 字符与字符代码

大多数系统采用**ASCII**字符集 [附录a]

◆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

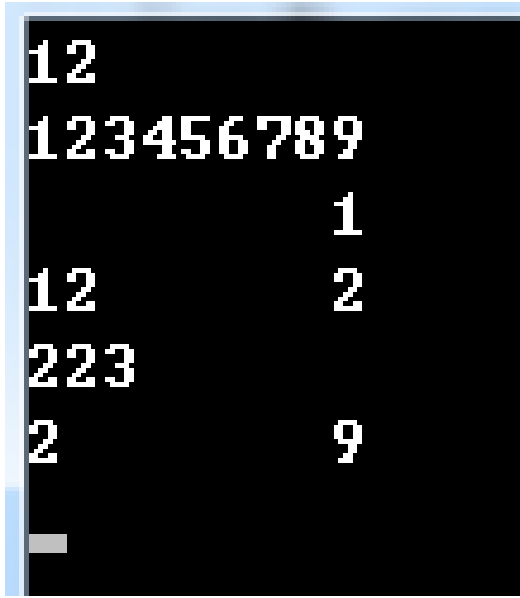
◆不能显示的字符: 空(**null**)字符(以'**\0**'表示)、警告(以'**\a**'表示)、退格(以'**\b**'表示)、回车(以'**\r**'表示)等





## 2.2.4 字符型数据

```
#include <stdio.h>
int main (void) {
 printf("12\r1\n");
 printf("123456789\n");
 printf("\t1\n");
 printf("12\t2\n");
 printf("123\r2\n");
 printf("123\r2\t9\n");
 return 0;
}
```



```
12
123456789
 1
12 2
223
2 9
-
```



## 2.2.4 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|



## 2.2.4 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 ?**



# 字符型数据与整型的关系

- 字符型数据实际上是用**1**个字节存储的整数。因此，其与**1**个字节范围内的整数是对应的。

## 2.2.4 字符型数据

### 3. 字符串常量

- 用双引号引起来的若干个字符
- 例如, **"a", "abc", "%dhd", "\n"**
- 注意区分字符串与字符（字符串的内存表示**p40**）
  - ◆ **'a'**与**"a"**不同！
  - ◆ **char c = 'a';** 正确    **char c = "a";** 错误



## 小测试

➤ 查附录**A**，写出程序结果

```
int main()
{
 char c1 = 'b';
 char c2 = 'c';
 printf("%d\n", c2-c1);
}
```

## 小测试

➤ 查附录**A**，写出程序结果

```
int main()
{
 char c1 = 'C';
 char c2 = 'c';
 printf("%d\n", c2-c1);
}
```

## 小测试

- 查附录**A**，区分'**0**'，**\0**，空格(**space**)这三个字符的区别



## 小测试

➤ 写程序实现如下功能：

输入一个小写字母，将其转成大写字母并输出。

## 2.2.5 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

◆规范化形式: **1.23e+002**

●**12.3e+001** (不是, 小数点前有且只有一个非零)

●**1.23e3.5** (不是, **e**后为整数)

◆数值以规范化的二进制数指数形式存放

参见主教材图**2.11**



## 2.2.5 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)

➤ **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



## 2.2.5 浮点型数据

- 关于整型数需要考虑表示范围，浮点数也需要考虑表示范围
- 此外，浮点数还需要考虑精度

```
int main()
{
 float a = 3.141592612;
 float b = a+0.000000001;
 printf("%.9f %.9f\n", a, b);
 return 0;
}
```

3.141592503 3.141592503



# 问题

➤ 如何判断浮点数相等？

◆ 请同学们思考？

## 2.2.6 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



## 小结

### ➤ 各种数据类型的概念

- ◆ 常量与变量的概念

- ◆ 变量名的命名规范

- ◆ **int(signed/unsigned, short, long), float, double, char, 字符串**

### ➤ 整数的内存表示

- ◆ 原码表示、补码表示

- ◆ 理解溢出概念

## 小结

- 理解**ASCII**码，理解**char**与**int**的关系
- 常用的几张表：
  - ◆ **P37 表2.2**（转义字符）
  - ◆ **附录A**



## 小结

- 理解实数的浮点表示方法（规范化浮点数）
- 理解**float**和**double**都是有精度限制的（从计算机的观点理解问题，而不是从数学观点理解），需要大致知道其表示范围和表示精度。

## 作业（自行完成，下次解答）

### ➤ 1、复习整数的补码表示，完成下列习题

- ◆ 用短整型数（内存表示为**2**个字节）存储十进制数**-32768**，请写出其补码形式。 [**1000 0000 0000 0000**]
- ◆ 用短整型数（内存表示为**2**个字节）存储十进制数**-1**，请写出其补码形式。 [**1111 1111 1111 1111**]
- ◆ 用短整型数（内存表示为**2**个字节）存储十进制数**32767**，请写出其补码形式。此数加**1**之后，结果为？ [**0111 1111 1111 1111**]+1
- ◆ 用无符号短整型数（内存表示为**2**个字节）存储十进制数**32767**，请写出其补码形式。此数加**1**之后，结果为？ [**0111 1111 1111 1111**]+1
- ◆ 用无符号短整型数（内存表示为**2**个字节）存储十进制数**65535**，请写出其补码形式。此数加**1**之后，结果为？ [**1111 1111 1111 1111**]+1

### ➤ 2、通过上机实验，写出你所在系统下**int**、**float**、**double**等数据类型可表示的最大数和最小数。

## 2.3 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



## 2.3 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

◆ 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**



## 2.3 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**



# 小测试

```
int i=0;
```

```
int k;
```

```
int j = i++;
```

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

```
k=j;
```

```
printf("%d", k);
```



## 2.3 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

**[重要！ 见附录！]**



## 2.3 运算符和表达式

### 4. 不同类型数据间的混合运算：[图2.12]

- (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





## 2.3 运算符和表达式

例 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**



## 2.3 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
```

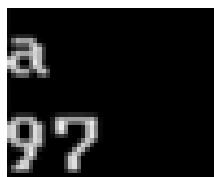
```
 c2=c1+32; 将65+32的结果放到c2中
```

```
 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



## 2.3 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



## 2.3 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 | ( ? : )              |



## 2.3 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 ( ) )  |



## 小结

### ➤ 运算符的含义

- ◆ `printf("%f", 5/4);`
- ◆ `printf("%f", 5/4.0);`
- ◆ `printf("%f", (float)(5/4));`
- ◆ `printf("%d", -2%3);`

### ➤ 运算符的优先级和结合性 [记忆附录]

- ◆ `int a = 5; int b; b = a++;` [优先级]
- ◆ `int a = 5, c = 4; int b = a = c;` [结合性]

## 2.4 C语句

### 2.4.1 C语句的作用和分类

### 2.4.2 最基本的语句-----赋值语句



## 2.4.1 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





## 2.4.2 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

◆ 输入输出语句

➤ 其中最基本的是赋值语句



## 2.4.2 最基本的语句----赋值语句

例**2.4** 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**)。

➤ 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



## 2.4.2 最基本的语句----赋值语句

```
#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
```



## ➤ 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;**



## 2.5 数据的输入输出

2.5.1 输入输出举例

2.5.2 有关数据输入输出的概念

2.5.3 用printf函数输出数据

2.5.4 用scanf函数输入数据

2.5.5 字符数据的输入输出



## 2.5.1 输入输出举例

例**2.5** 求  $ax^2 + bx + c = 0$  方程的根。

**a、b、c**由键盘输入

设  $b^2 - 4ac > 0$



## 2.5.1 输入输出举例

- 解题思路：首先要知道求方程式的根的方法。
- 由数学知识已知：如果  $b^2 - 4ac \geq 0$ ，则一元二次方程有两个实根：

$$x_1 = \frac{-b + \sqrt{b^2 - 4ac}}{2a} \quad x_2 = \frac{-b - \sqrt{b^2 - 4ac}}{2a}$$

若记  $p = \frac{-b}{2a} \quad q = \frac{\sqrt{b^2 - 4ac}}{2a}$

$$x_1 = p + q$$
$$x_2 = p - q$$



# C程序设计

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

```
#include <math.h> 程序中调用数学函数sqrt
```

```
int main ()
```

```
{double a,b,c,disc,x1,x2,p,q;
```

```
scanf("%lf%lf%lf",&a,&b,&c);
```

```
disc=b*b-4*a*c; 输入a,b,c的值
```

```
p=-b/(2.0*a);
```

```
q=sqrt(disc)/(2.0*a);
```

```
x1=p+q; x2=p-q;
```

```
printf("x1=%7.2f\nx2=%7.2f\n",x1,x2);
```

```
return 0;
```

```
}
```



## C程序设计

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

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

```
int main ()
```

```
{double a,b,c,disc,x1,x2,p,q;
```

```
scanf("%lf%lf%lf",&a,&b,&c);
```

```
disc=b*b-4*a*c;
```

```
p=-b/(2.0*a);
```

```
q=sqrt(disc)/(2.0*a);
```

```
x1=p+q; x2=p-q;
```

```
printf("x1=%7.2f\nx2=%7.2f\n",x1,x2);
```

```
return 0;
```

```
}
```

输入的是双  
精度型实数



# C程序设计

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

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

```
int main ()
```

```
{double a,b,c,disc,x1,x2,p,q;
```

```
scanf("%lf%lf%lf",&a,&b,&c);
```

```
disc=b*b-4*a*c;
```

```
p=-b/(2.0*a);
```

```
q=sqrt(disc)/(2.0*a);
```

```
x1=p+q; x2=p-q;
```

```
printf("x1=%7.2f\nx2=%7.2f\n",x1,x2);
```

```
return 0;
```

```
}
```

1 3 2

自动转成实数  
后赋给a,b,c

要求输入3个实数





# C程序设计

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

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

```
int main ()
```

```
{double a,b,c,disc,x1,x2,p,q;
```

```
scanf("%lf%lf%lf",&a,&b,&c);
```

```
disc=b*b-4*a*c;
```

```
p=-b/(2.0*a);
```

```
q=sqrt(disc)/(2.0*a);
```

```
x1=p+q; x2=p-q;
```

```
printf("x1=%7.2f\nx2=%7.2f\n",x1,x2);
```

```
return 0;
```

```
}
```

```
1 3 2
x1= -1.00
x2= -2.00
```

输出数据占7列，其中小数占2列



## 2.5.2 有关数据输入输出的概念

- 几乎每一个**C**程序都包含输入输出
- 输入输出是程序中最基本的操作之一



## 2.5.2 有关数据输入输出的概念

**(1)** 所谓输入输出是以计算机主机为主体而言的

- 从计算机向输出设备(如显示器、打印机等)输出数据称为输出
- 从输入设备(如键盘、磁盘、光盘、扫描仪等)向计算机输入数据称为输入



## 2.5.2 有关数据输入输出的概念

(2) C语言本身不提供输入输出语句

- 输入和输出操作是由C标准函数库中的函数来实现的
- **printf**和**scanf**不是C语言的关键字，而只是库函数的名字
- **putchar**、**getchar**、**puts**、**gets**



## 2.5.2 有关数据输入输出的概念

**(3)** 在使用输入输出函数时，要在程序文件的开头用预编译指令

**#include <stdio.h>**

或

**#include "stdio.h"**



## 2.5.3 用printf函数输出数据

- 在C程序中用来实现输出和输入的，主要是**printf**函数和**scanf**函数
- 这两个函数是格式输入输出函数
- 用这两个函数时，必须指定格式



## 2.5.3 用printf函数输出数据

### 1.printf函数的一般格式

**printf**（格式控制，输出表列）

例如：

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

格式声明



## 2.5.3 用printf函数输出数据

### 1.printf函数的一般格式

**printf**（格式控制，输出表列）

例如：

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

普通字符





## 2.5.3 用printf函数输出数据

### 1.printf函数的一般格式

**printf**（格式控制，输出表列）

例如：

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

可以是常量、变量或表达式



## 2.5.3 用printf函数输出数据

### 2. 常用格式字符

◆ d 格式符。用来输出一个有符号的十进制整数

●可以在格式声明中指定输出数据的域宽

```
printf("%5d%5d\n",12,-345);
```

●**%d**输出**int**型数据

●**%ld**输出**long**型数据



## 2.5.3 用printf函数输出数据

### 2. 常用格式字符

◆ c 格式符。用来输出一个字符

```
char ch='a';
```

```
printf("%c",ch); 或
```

```
printf("%5c",ch);
```

输出字符: a



## 2.5.3 用printf函数输出数据

### 2. 常用格式字符

◆ s 格式符。用来输出一个字符串

```
printf ("%s","CHINA");
```

输出字符串: **CHINA**



## 2.5.3 用printf函数输出数据

### 2. 常用格式字符

◆**f**格式符。用来输出实数，以小数形式输出

①不指定数据宽度和小数位数，用**%f**

例**3.6** 用**%f**输出实数，只能得到 6 位小数。

```
double a=1.0;
```

```
printf("%f\n",a/3);
```

```
0.333333
```



## 2.5.3 用printf函数输出数据

### 2. 常用格式字符

◆**f**格式符。用来输出实数，以小数形式输出

② 指定数据宽度和小数位数。用**%m.nf**

```
printf("%20.15f\n",1/3);
```

```
0.333333333333333
```

```
printf("%.0f\n",10000/3.0);
```

```
3333
```



## 2.5.3 用printf函数输出数据

### 2. 常用格式字符

◆**f**格式符。用来输出实数，以小数形式输出

② 指定数据宽度和小数位数。用**%m.nf**

```
float a;
```

```
a=10000/3.0;
```

```
printf("%f\n",a);
```

```
3333.333333
```



## 2.5.3 用printf函数输出数据

### 2. 常用格式字符

◆**f**格式符。用来输出实数，以小数形式输出

③ 输出的数据向左对齐，用**%-m.nf**





## 2.5.3 用printf函数输出数据

### 2. 常用格式字符

- ◆**f**格式符。用来输出实数，以小数形式输出
  - float**型数据只能保证**6**位有效数字
  - double**型数据能保证**15**位有效数字
  - 计算机输出的数字不都是绝对精确有效的



## 2.5.3 用printf函数输出数据

### 2. 常用格式字符

◆**e**格式符。指定以指数形式输出实数

●**%e**，**VC++**给出小数位数为6位

指数部分占**5**列

小数点前必须有而且只有**1**位非零数字

```
printf("%e",123.456);
```

输出: **1.234560 e+002**



## 2.5.3 用printf函数输出数据

### 2. 常用格式字符

◆**e**格式符。指定以指数形式输出实数

●**%m.ne**

```
printf("%13.2e",123.456);
```

输出:      **1.23e+002**      (前面有**4**个空格)



## 2.5.4 用scanf函数输入数据

### 1. scanf 函数的一般形式 ■

**scanf** (格式控制, 地址表列)

含义同printf函数



## 2.5.4 用scanf函数输入数据

### 1. scanf 函数的一般形式 ■

**scanf**（格式控制，地址表列）

可以是变量的地址，或字符串的首地址



## 2.5.4 用scanf函数输入数据

### 2. scanf函数中的格式声明 ■

- 与**printf**函数中的格式声明相似
- 以%开始，以一个格式字符结束，中间可以插入附加的字符

```
scanf("a=%f,b=%f,c=%f",&a,&b,&c);
```



## 2.5.4 用scanf函数输入数据

### 3. 使用scanf函数时应注意的问题 ■

`scanf("%f%f%f",a,b,c);` 错

`scanf("%f%f%f",&a,&b,&c);` 对

对于

`scanf("a=%f,b=%f,c=%f",&a,&b,&c);`

1 3 2 ✓

错

a=1,b=3,c=2 ✓

对

a=1 b=3 c=2 ✓

错



## 2.5.4 用scanf函数输入数据

### 3. 使用scanf函数时应注意的问题 ■

对于scanf("%c%c%c",&c1,&c2,&c3);

abc✓

对

a b c✓

错

对于scanf("%d%c%f",&a,&b,&c);

若输入

1234a123o.26✓





## 2.5.4 用scanf函数输入数据

### 3. 使用scanf函数时应注意的问题 ■

对于scanf("%c%c%c",&c1,&c2,&c3);

abc✓

对

a b c✓

错

对于scanf("%d%c%f",&a,&b,&c);

若输入

1234a123o.26✓



## 2.5.4 用scanf函数输入数据

### 3. 使用scanf函数时应注意的问题 ■

对于scanf("%c%c%c",&c1,&c2,&c3);

abc ✓ 对

a b c ✓ 错

对于scanf("%d%c%f",&a,&b,&c);

若输入

1234a123o.26 ✓



## 2.5.5 字符数据的输入输出

### 1. 用**putchar**函数输出一个字符

- 从计算机向显示器输出一个字符
- **putchar**函数的一般形式为：

**putchar(c)**



## 2.5.5 字符数据的输入输出

例**3.8** 先后输出**BOY**三个字符。

➤ 解题思路：

- ◆ 定义**3**个字符变量，分别赋以初值**B**、**O**、**Y**
- ◆ 用**putchar**函数输出这**3**个字符变量的值



## 2.5.5 字符数据的输入输出

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

```
int main ()
```

```
{
```

```
 char a='B',b='O',c='Y';
```

```
 putchar(a); 向显示器输出字符B
```

```
 putchar(b);
```

```
 putchar(c);
```

```
 putchar ('\n'); 向显示器输出换行符
```

```
 return 0;
```

```
}
```

BOY



## 2.5.5 字符数据的输入输出

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

```
int main ()
```

改为int a=66,b=79,c=89;

```
{
```

```
 char a='B',b='O',c='Y';
```

```
 putchar(a);
```

```
 putchar(b);
```

```
 putchar(c);
```

```
 putchar ('\n');
```

```
 return 0;
```

```
}
```

BOY



## 2.5.5 字符数据的输入输出

**putchar** ( '\101' )      (输出字符 A )

**putchar** ( '\"' )      (输出单撇号字符')



## 2.5.5 字符数据的输入输出

### 2. 用**getchar**函数输入一个字符

- 向计算机输入一个字符
- **getchar**函数的一般形式为：

**getchar( )**





## 2.5.5 字符数据的输入输出

例**3.9** 从键盘输入**BOY**三个字符，然后把它们输出到屏幕。

➤ 解题思路：

- ◆ 用**3**个**getchar**函数先后从键盘向计算机输入**BOY**三个字符
- ◆ 用**putchar**函数输出



## 2.5.5 字符数据的输入输出

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

```
int main ()
```

```
{ char a,b,c;
```

```
 a=getchar(); 输入一个字符，送给变量a
```

```
 b=getchar();
```

```
 c=getchar();
```

```
 putchar(a); putchar(b); putchar(c);
```

```
 putchar('\n');
```

```
 return 0;
```

```
}
```



## 2.5.5 字符数据的输入输出

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

```
int main ()
```

```
{ char a,b,c;
```

```
 a=getchar(); putchar(getchar());
```

```
 b=getchar();
```

```
 c=getchar();
```

```
 putchar(a); putchar(b); putchar(c);
```

```
 putchar('\n');
```

```
 return 0;
```

```
}
```



## 2.5.5 字符数据的输入输出

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

```
int main ()
```

```
{ char a,b,c;
```

```
 putchar(getchar());
```

```
 b=getchar();
```

```
 c=getchar();
```

```
 putchar(b); putchar(c);
```

```
 putchar('\n');
```

```
 return 0;
```

```
}
```



## 2.5.5 字符数据的输入输出

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

```
int main ()
```

```
{ char a,b,c;
```

```
 c=getchar();
```

```
 putchar(getchar());
```

```
 putchar(getchar());
```

```
 putchar(getchar());
```

```
 putchar(c);
```

```
 putchar('\n');
```

```
 return 0;
```

```
}
```



## 2.5.5 字符数据的输入输出

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

```
int main ()
```

```
{ char a,b,c;
```

```
 putchar(getchar());
```

```
 putchar(getchar());
```

```
 putchar(getchar());
```

```
 putchar('\n');
```

```
 return 0;
```

```
}
```



BOY  
BOY



## 小测试

➤ 输出十进制**97**对应的**ASCII**码字符

◆ **printf("%c", 97);**

◆ **putchar(97);**

➤ 输出**"hello world!"**

◆ **printf("\\\"hello world!\\");**

## 本次作业

- 复习本章内容 预习第三章
- 课后习题**2.1, 2.2, 2.3, 2.4**  
(自行完成, 下次选讲)
- **OJ**试题 (周日晚上**23:59**前提交)