



浙江省“十一五”重点建设教材

高等院校规划教材

C语言程序设计

实践教程

江宝钊 主 编

杨任尔 宋宝安 副主编



清华大学出版社



浙江省“十一五”重点建设教材

高等院校规划教材

C语言程序设计

实践教程

江宝钊 主 编

杨任尔 宋宝安 副主编

常州大学图书馆
藏书章

清华大学出版社
北 京

内 容 简 介

本书是与清华大学出版社出版的《C 语言程序设计》(江宝钊主编)配套的上机实践指导书。

全书共分 11 章。前 10 章每章根据需要设有知识要点、常见错误和难点分析、程序验证与调试、测试与练习以及实验项目等模块。每个实验项目都以知识点为主线,从易到难精心设计了程序验证题与改错题、程序修改与扩展题、一般编程题和较难编程题。最后一章是经典例题分析,按实际问题分类兼具一定的趣味性、实用性和经典性,有利于学生进一步理解和掌握编程方法。

本书适合作为各类高等院校计算机专业及理工类非计算机专业的 C 语言程序设计课程的实验教材。

本书封面贴有清华大学出版社防伪标签,无标签者不得销售。

版权所有,侵权必究。侵权举报电话:010-62782989 13701121933

图书在版编目(CIP)数据

C 语言程序设计实践教程/江宝钊主编. —北京:清华大学出版社,2012.1

(高等院校规划教材)

ISBN 978-7-302-27696-8

I. ①C… II. ①江… III. ①C 语言—程序设计—高等学校—教学参考资料
IV. ①TP312

中国版本图书馆 CIP 数据核字(2011)第 268172 号

责任编辑:孟毅新

责任校对:袁 芳

责任印制:杨 艳

出版发行:清华大学出版社

地 址:北京清华大学学研大厦 A 座

<http://www.tup.com.cn>

邮 编:100084

社 总 机:010-62770175

邮 购:010-62786544

投稿与读者服务:010-62776969, c-service@tup.tsinghua.edu.cn

质 量 反 馈:010-62772015, zhiliang@tup.tsinghua.edu.cn

印 刷 者:北京富博印刷有限公司

装 订 者:北京市密云县京文制本装订厂

经 销:全国新华书店

开 本:185×260 印 张:8 字 数:192 千字

版 次:2012 年 1 月第 1 版 印 次:2012 年 1 月第 1 次印刷

印 数:1~3000

定 价:16.00 元

产品编号:045730-01

前 言

C 语言程序设计是一门对实践环节要求很高的课程。要学好这门课程,重要的是要抓住两个关键点:多做习题,多编程;反复上机调试、验证,也就是学习者必须通过大量反复的编程训练,在实践中掌握程序设计的方法,培养程序设计的基本能力。本书的设计思路是以循序渐进的方式引导学生提高自身的实践编程能力。

本书共分 11 章,每章根据需要设有知识要点、常见错误和难点分析、程序验证与调试、测试与练习以及实验项目等模块。“知识要点”部分根据教材相应章节概括了应该掌握的知识点。教师多年的 C 语言教学经验和上机辅导中发现的问题则放在“常见错误和难点分析”中。每章都提供了程序调试示例,或程序验证与程序改错,旨在加强学生的“模仿与调试修改”的能力。每章还针对性地提供了大量的练习题和编程题;实验项目内容包括编程修改与扩展、一般编程题和较难编程题,教师可根据学生编程的情况做出调整,这些内容旨在提高学生的“编程”能力。

本书的最后一章是经典例题分析,结合经典问题或实际问题分成多个类别,每个类别都提供了数个兼具一定趣味性、实用性和经典性的例题及编程分析,促使学生进一步理解和掌握编程方法,激发学生的学习兴趣。

本书由宁波大学信息学院长期工作在教学一线的教师编写。第 1~3 章、第 11 章由江宝钏编写,第 4~7 章由杨任尔编写,第 8~10 章由宋宝安编写。本书由江宝钏担任主编。

本书的编写得到了浙江省重点教材建设项目经费和宁波大学教材建设项目经费的资助,陈叶芳、李纲、董一鸿、沈明忻、邬延辉、石守东、傅振宇等多位任课老师参与了本书配套教材的前期编写工作,他们对本书的大纲和实验内容提出了许多修改意见和建议,在此表示衷心的感谢!

本书经过多次反复修改讨论而定稿,限于编者的水平,书中难免有不足之处,恳请读者批评指正,以便进一步修订。

作者联系地址:浙江省宁波大学信息科学与工程学院,邮编:315211; E-mail:jiangbaochuan@nbu.edu.cn。

编 者

2011 年 11 月

目 录

第 1 章 VC++ 6.0 编程环境与编程基础	1
1.1 知识要点	1
1.2 在 VC++ 6.0 环境下运行 C 语言程序	2
实验一 熟悉 C 语言的 VC++ 6.0 编程环境	5
第 2 章 顺序结构程序设计	9
2.1 知识要点	9
2.2 常见错误和难点分析	11
2.3 程序调试初步	12
2.4 测试与练习	14
实验二 编写几个简单的 C 语言程序	15
第 3 章 选择结构程序设计	18
3.1 知识要点	18
3.2 常见错误和难点分析	19
3.3 程序验证与调试	21
3.4 测试与练习	27
实验三 选择结构程序设计	30
第 4 章 循环结构程序设计	33
4.1 知识要点	33
4.2 常见错误和难点分析	34
4.3 程序调试	35
4.4 测试与练习	38
实验四 循环结构程序设计	42
第 5 章 函数与预处理	44
5.1 知识要点	44
5.2 常见错误和难点分析	47
5.3 测试与练习	50
实验五 函数与预处理	54
第 6 章 数组	58
6.1 知识要点	58
6.2 常见错误和难点分析	59

6.3 测试与练习	61
实验六 数组	65
第 7 章 字符与字符串处理	67
7.1 知识要点	67
7.2 常见错误和难点分析	69
7.3 测试与练习	71
实验七 字符与字符串处理	74
第 8 章 指针	77
8.1 知识要点	77
8.2 常见错误和难点分析	79
8.3 测试与练习	80
实验八 指针	81
第 9 章 结构体与链表	84
9.1 知识要点	84
9.2 常见错误和难点分析	85
9.3 测试与练习	87
实验九 结构体与链表	89
第 10 章 文件	92
10.1 知识要点	92
10.2 常见错误和难点分析	92
10.3 测试与练习	93
实验十 文件操作	94
第 11 章 经典例题分析	97
11.1 选择结构	97
11.2 数列求和	100
11.3 组合与穷举问题求解	101
11.4 最大值、最小值和最大公约数、最小公倍数的求解	103
11.5 数制转换	105
11.6 寻找水仙花数、完数、同构数与亲密数对	107
11.7 素数问题	109
11.8 字符串处理	110
11.9 方程根、平方根的求解与数值的迭代	114
11.10 综合应用	117
参考文献	121

第 1 章 VC++ 6.0 编程环境与编程基础

1.1 知识要点

1. C 语言程序的标识符

标识符是给程序中的实体——变量、常量、函数、数组、结构体以及文件等所起的名字。

C 语言中的标识符分为系统定义标识符和用户定义标识符。

系统定义标识符是指具有固定名字和特定含义的保留标识符,如 int、for、break 等。

用户在定义标识符时应符合 C 语言中标识符的命名规则。

(1) 标识符以字母或下画线开头,由字母、数字和下画线 3 种字符组成。

(2) C 语言将大写字母和小写字母作为不同的字符处理。如系统认为 sum、Sum、suM、SUM 是不同的标识符。

(3) 标识符不能与 C 语言编译系统中已经预定义、具有特殊用途的保留标识符(关键字)同名。

2. C 语言程序的基本组成结构

C 语言程序由函数、编译预处理命令及注释 3 部分组成。

一个完整的 C 语言程序可以由一个或多个函数组成,其中,主函数 main()必不可少,且只能唯一。C 语言程序执行时,总是从主函数 main()开始,与 main()函数在整个程序中的位置无关。函数分为系统提供的库函数和用户自定义的函数。

C 语言程序的基本结构如下:

```
#include <stdio.h>          /* 编译预处理命令 */
void main()
{
    定义部分                /* 这是程序的定义部分 */
    语句序列                // 这是程序的语句部分
}
```

(1) 编译预处理命令

程序中每一个以“#”号开头的命令行,是编译预处理命令,一般放在程序的最前面。不同的编译预处理命令完成不同功能。如 #include <stdio.h> 命令的作用是将特定目录下的 stdio.h 文件,嵌入源程序中。

(2) 函数

① 函数说明,如 int average (float x,float y),包括函数类型、函数名、参数类型、函数参数。

② 函数体,是函数首行下面花括号对中的内容。函数体由各类语句组成,执行时按语句的先后次序依次执行,各语句用分号“;”结束。

(3) 注释

“/* 注释内容 */”适用于注释多行,/* 和 */之间的内容即为注释。

“// 注释内容”适用于注释单行, //后面的部分即为注释。

(4) C 语言程序的输入与输出

C 语言程序的输入与输出通过库函数 `scanf()` 和 `printf()` 实现。

3. C 语言程序的运行

C 语言程序的运行必须经过编辑源程序、编译和连接及调试运行, 最后得到可执行程序。

1.2 在 VC++ 6.0 环境下运行 C 语言程序

1. 启动 Visual C++ 6.0 集成开发环境

首先建立以自己学号为名称的文件夹, 如“E:\×××”。选择“开始”→“程序”→Microsoft Visual Studio 6.0→Microsoft Visual C++ 6.0 命令, 进入 VC++ 编程环境, 如图 1.1 所示。

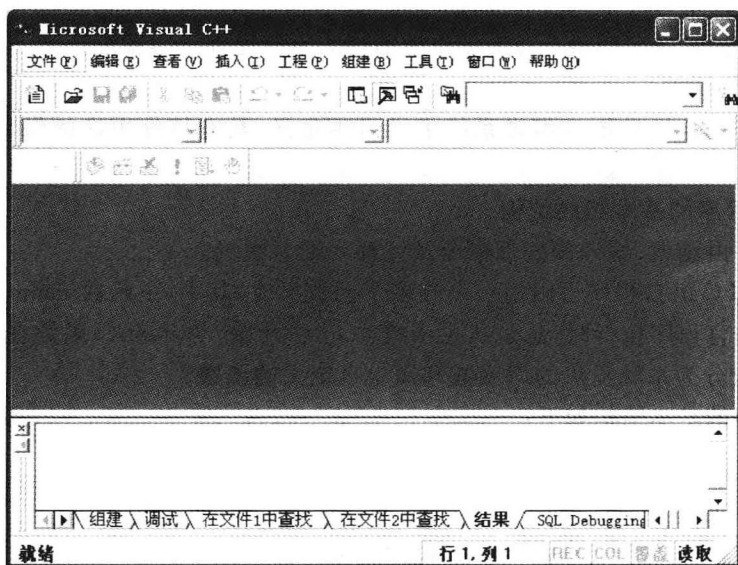


图 1.1 VC++ 编程环境

2. 新建文件

选择“文件”→“新建”命令, 弹出“新建”对话框, 选择“文件”选项卡, 如图 1.2 所示。

首先在“新建”对话框右侧的“文件名”文本框内输入 `expl_1`, 即把源程序文件命名为 `expl_1.cpp`, 在“位置”下拉列表框中选择要保存文件的文件夹, 如 `E:\09601001`; 然后选中 `C++ Source File` 选项, 单击“确定”按钮, 在 `E:\09601001` 下建立了新建文件 `expl_1.cpp`, 并显示编辑窗口和信息窗口, 如图 1.3 所示。

3. 程序的编辑、编译、运行

(1) 程序的编辑和保存

在图 1.3 的编辑窗口中输入如下源程序, 然后执行“文件”→“保存”命令或单击  按钮保存源程序。

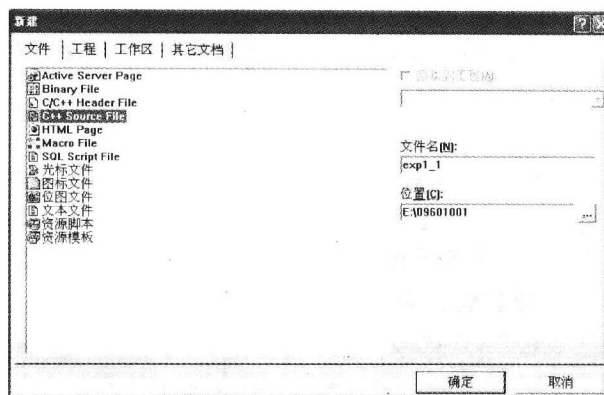


图 1.2 “新建”对话框

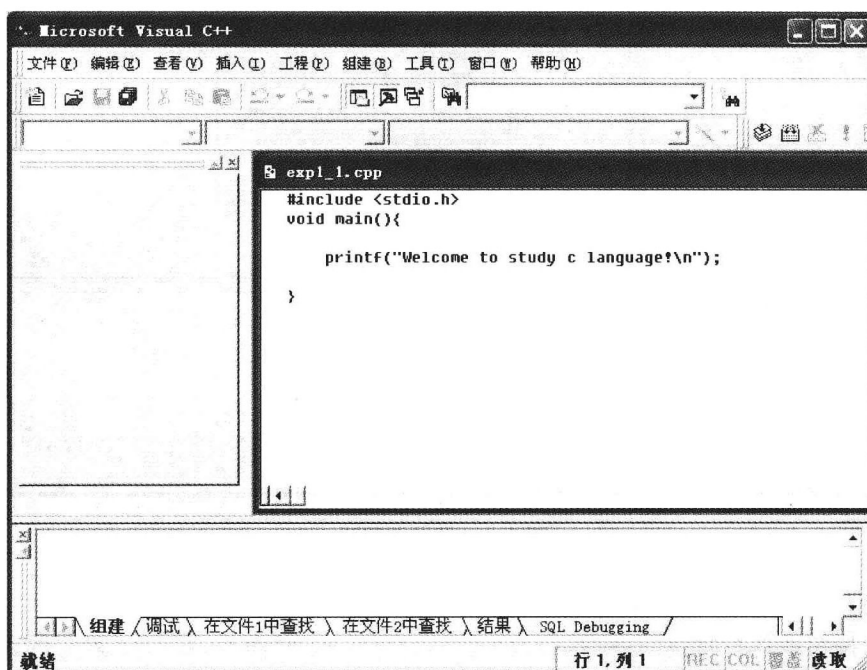


图 1.3 编辑源程序

```
#include <stdio.h>
void main()
{
    printf("Welcome to study c language!\n");
}
```

将上述内容保存为 expl_1.cpp。

(2) 程序的编译

执行“组建”→“编译 expl_1.cpp”命令,即可编译源文件 expl_1.cpp,假如程序编译有错,系统会在下方 Output 窗口显出错误(Error)信息以及警告(Warning)信息。当所有

Error 改正后,可得到目标文件 `expl_1.obj`。如图 1.4 所示,下方信息窗口中出现的“`myproject1_1.exe - 0 error(s),0 warning(s)`”表示编译正确,没有发现错误和警告,并生成了目标文件 `expl_1.obj`。

(3) 程序的连接

在图 1.4 中,执行“组建”→“组建 `myproject1_1.exe`”命令,连接并建立工程的 `exe` 文件,得到可执行文件 `expl_1.exe`。这时编译器可能会给出连接错误(Linking Error)。产生连接错误的原因可能是缺少所需要的库文件或目标文件,或程序中调用的外部函数没有定义等,只要补充相应文档再重新建立即可。

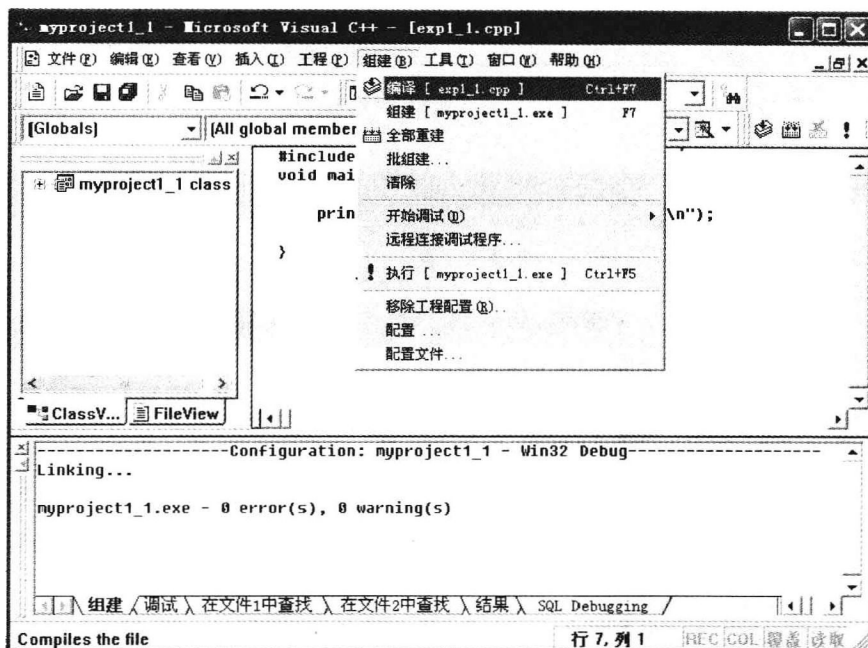


图 1.4 编译源程序

(4) 程序的运行

单击! 按钮或执行“组建”→“执行[`myproject1_1.exe`]”命令,执行工程文件,会出现一个类似 DOS 操作系统的窗口,显示运行结果“Welcome to study C language!”,其中 Press any key to continue 表示用户按任意键退出运行窗口返回到 VC 编辑窗口,如图 1.5 所示。

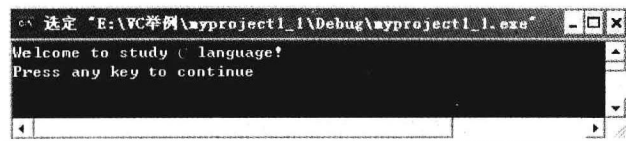


图 1.5 `expl-1` 运行结果

4. 关闭和打开程序工作区

执行“文件”→“关闭工作空间”命令或执行“文件”→“打开工作空间”命令。

5. 查询和打开 C 语言程序的文件

源程序经过编辑、编译、连接和运行后,源文件 `expl_1.cpp` 保存在前面新建的文件夹中,目标文件 `expl_1.obj` 和可执行文件 `expl_1.exe` 保存在所创建的文件夹下的 Debug 子文件夹中。执行“文件”→“打开”命令,在自己创建的文件夹中选择文件 `expl_1.cpp` 可打开源文件。

实验一 熟悉 C 语言的 VC++ 6.0 编程环境

一、实验目的与要求

1. 学会使用 Visual C++ 集成开发环境。
2. 掌握简单 C 语言程序的编写、编辑、编译、连接和运行。
3. 初步学会 Visual C++ 环境下简单调试 C 语言程序的方法。

二、实验内容

1. 程序运行题

- (1) 打开 VC++ 6.0 环境。
- (2) 在 VC++ 6.0 环境的编辑窗口中,输入如下程序保存为 `myexpl_1.cpp`。
- (3) 执行“组建”→“编译 `myexpl_1.cpp`”(可按快捷键 `Ctrl+F7`)命令对程序进行编译。
- (4) 执行“组建”→“组建”(按 `F7` 键)命令对程序进行连接。
- (5) 单击!按钮或执行“组建”→“执行”命令(按 `F5` 键),运行程序。
- (6) 判断结果正确与否。

```
#include <stdio.h>
void main()
{
    int a, b, sum;
    a = 123;
    b = 456;
    sum = a + b;
    printf("a + b = %d\n", sum);
}
```

运行结果:

关闭上面的程序,按上面的步骤再次输入如下的程序,进行编译、连接和运行,写出运行结果。

```
#include <stdio.h>
void main()
{
    int m = 7, n = 4;
    float a = 38.4, b = 6.4, x;
    x = m/2 + n * a/b + 1/2;
```

```
printf("%f\n", x);
}
```

运行结果:

思考: 把前面的语句“ $x=m/2+n*a/b+1/2$;”改为“ $x=m/2.0+n*a/b+1.0/2$;”, 运行结果将会是什么?

2. 程序改错题

(1) ① 输入如下错误的程序, 然后进行编译和连接, 仔细检查并修改错误, 再进行编译连接, 直到没有错误为止。

```
#include <stdio.h>
void mian()
{
    printf("We are studying C language!\n")
}
```

② 运行程序。

(2) 输入并运行下面一个需要在运行时输入数据的程序, 程序要求从键盘输入 3 个实数, 输出它们的和。

```
#include <stdio.h>
mian()
{
    float a,b,c,sum;
    printf(" Input a,b,c: ")
    scanf("%f %f %f",&a,&b, c);
    sum = a + b + c;
    printf("\n sum = %f\n",sum);
}
```

① 输入该程序, 然后进行编译和连接, 如果出错, 仔细检查并修改错误, 再进行编译连接, 直到没有错误为止。

② 运行程序, 按照输入、输出示例提示输入 a、b、c 三个变量的值, 变量值之间用空格分隔, 输入完毕按 Enter 键。

③ 查看运行结果, 判断结果正确与否。

输入、输出示例:

```
Input a,b,c: 3 5 6↵
sum = 14.0
```

再次运行程序并按如下的格式输入数据, 观察结果是否正确? 为什么?

```
Input a,b,c: 3,5,6↵
sum = ?
```

思考: 改正程序的错误后, 把 printf 语句中的 %f 改成 %d, 再运行程序, 有什么结果?

Input a,b,c: 3.5,5.5,6.5 ✓
结果:

3. 程序完整题

(1) 两整数整除程序模板如下:

```
#include <stdio.h>
void main()
{
    int a,b, div;
    printf(" Input a,b: ");          /* 提示输入两个数 */
    scanf("%d%d",&a,&b);             /* 从键盘输入两个数存储在变量 a,b 中 */

    _____ ① _____         /* 除法运算,相除的结果放在变量 div 中 */

    _____ ② _____         /* 用 %d 输出结果 */
}
```

输入、输出示例如下:

Input a,b: 3 5 ✓
div = 0

Input a,b: 5 3 ✓
div = 1

思考: 把上述“int a,b,div;”改成“float a,b,div;”,将“scanf(“%d%d”,&a,&b);”中的%d改成%f,同时把输出语句中的%d改成%f,运行程序后,再用前面的数据输入,结果会如何变化? 为什么有这样的变化?

Input a,b: 3,5 ✓
div =

Input a,b: 5,3 ✓
div =

(2) 仿照教材中例 1-2 程序,在横线空白处填写语句,输入圆柱体的半径和高,计算并输出圆柱体的体积和表面积。

```
#include <stdio.h>
void main()
{
    _____ ① _____;          /* 定义变量 */
    printf("请输入圆柱体的半径和高: \n"); /* 在屏幕上显示提示信息 */
    scanf("%f%f", &r,&h);             /* 从键盘输入半径和高给变量 r */
    _____ ② _____;          /* 计算体积 */
    _____ ③ _____;          /* 计算表面积 */

    printf("v = %f, s = %f\n", v,s);    /* 输出体积和表面积 */
}
```

输入、输出示例如下：

请输入圆柱体的半径和高：↵
5.0 8.0 ↵
输出结果？

思考：

① 如果把上述程序中所有的%f改成%d,再次运行程序,仍然输入“5.0 8.0”,输出结果会是什么? 分析结果。

② 把上述程序中的第一个\n去掉,再编译运行程序,有什么变化? 试说出\n的作用。

第2章 顺序结构程序设计

2.1 知识要点

1. 基本数据类型

不同类型的数据在内存中所占的存储空间不同,取值范围不同,对应的运算也不同。每一个数据都属于唯一的一种数据类型。C语言的数据类型如主教材中的图 2-1 所示。

基本数据类型的数据表示和取值范围(VC++ 6.0 系统)如主教材中的表 2-1 所示。

2. 常量与符号常量

C语言中的常量分为常量和符号常量。

常量是指在程序运行过程中其值不可改变的量。常量分为字符常量、整型常量、实型常量、字符串常量。

符号常量指的是用某个标识符来代替程序中经常用到的常量。使用符号常量可以提高程序的可读性和可移植性,符号常量必须先定义后使用。

整型常量即整常数。有十进制整数、八进制整数和十六进制整数 3 种类型。八进制整数,必须以 0 开头,由数字 0~7 组成;十六进制整数,必须以 0x 或 0X 开头。

在一个整型常量后面加一个字母 l 或 L,则认为是长整型常量,例如 224l、-34L 等。

实型常量就是实数。实型常量有两种表示形式,小数形式和指数形式。

注意: 小数形式必须有小数点;指数形式字母 E 之前必须有数字,且 E 后面的指数必须为整数,如 E4、3.4E2.5、.E4 等都不是合法的指数形式。

字符常量是用英文单引号括起来的一个字符。如'a'、'A'、'B'、'#'、'%'等都是字符常量。

注意: 'a'和'A'是不同的字符常量。

转义字符是特殊的字符常量,指的是以一个“\”字符开头的字符序列,用于表示 ASCII 字符集内的控制代码和某些功能定义,例如,换行、回车等。

(1) 转义字符代表一个字符,在内存中只占一个字节。

(2) 转义字符\0 就是 ASCII 值为 0 的字符,用于表示字符串常量结束的标志。

(3) 转义字符\101 代表 ASCII 码(八进制数)为 101 的字母 A。

字符串常量是由一对英文双引号括起的字符序列。例如"ZHEJIANG"、"C programv"。

3. 变量

变量是在程序执行中其值可以变化的量,在内存中有相应的存储单元。每个变量都具有名字、类型和值,必须先进行定义。

注意: 数据类型可以是主教材图 2-1 中任意一种基本数据类型或构造类型标识符,可以同时定义多个变量,习惯上用小写字母命名。

变量在使用之前需要赋给初值,如果没有给它赋初值,变量值是一个不确定的数据,直

接使用它没有意义。例如：

```
int a;           /* 定义整型变量 a */
a = 10;          /* a 的值为 10 */
```

C 语言中可以在定义变量的同时为其赋值,称为初始化。变量可以全部或部分初始化。例如：

```
int x = 14, y = 7; /* 对变量 x 和 y 进行了初始化 */
int a = 4, b, c = 6; /* 只对变量 a 和 c 进行了初始化 */
```

注意：在变量定义中不允许连续赋值。例如：

```
int a = b = c = 6; /* 错误,因为变量连续赋值 */
```

== 是赋值符号,不是等于号。在 C 语言中等于号用 == 表示。例如：

```
a = 10;           /* 该语句是对变量 a 进行赋值 */
a == 10;          /* 该语句是判断变量 a 的值是否等于 10 */
```

4. 常用运算符

(1) 除法运算符/,当两个操作数都是整数时,运算的结果是整数;如果参加运算的两个数中有一个是实数,则结果是双精度实型数据。

(2) 求余运算符%,又称取模运算符,要求运算对象必须为整型数,它的作用是求两个整型数相除的余数,余数的符号与被除数的符号相同。

(3) 自增运算符++和自减运算符--是单目运算符,而且运算对象必须是变量,不能是常量。结合方向是自右向左,作用是使运算对象的值加 1 或减 1。在只需对变量本身进行加 1 或减 1 而不考虑表达式的情况下,前缀运算和后缀运算的效果完全相同,否则,结果是有区别的。

5. 复合的赋值表达式

在赋值符=之前加上其他运算符,可以构成复合的运算符。如果在=前加一个-运算符就成了复合运算符-=。赋值运算符具有右结合性。

6. 逗号运算符和逗号表达式

C 语言中逗号“,”也是一种运算符,是把多个表达式连接起来组成一个表达式。例如：

表达式 1, 表达式 2, 表达式 3, ..., 表达式 n

先求解表达式 1,再求解表达式 2,整个逗号表达式的值是表达式 n 的值。

7. 各类数据的数据混合运算

当运算对象是不同类型的数据时,系统自动将操作数转换成相同的类型后再进行计算。

转换的方法有两种：自动类型转换和强制类型转换。

(1) 自动类型转换

在 C 语言中,允许不同类型的数据进行混合运算,先自动进行必要的数据类型转换,然后再进行运算转换。转换规则如图 2.1 所示。

在赋值运算中,当=运算符两边的运算对象数据类型

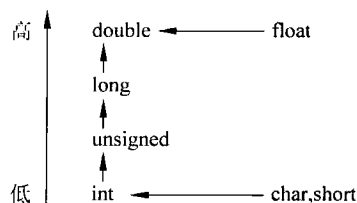


图 2.1 自动类型转换的规则

不一致时,系统自动将=运算符右边的表达式的值转换成左边的变量类型后再赋值。例如:

```
int x = 11.4;           /* 变量 x 的值是 11 */
long y = 97;            /* 变量 y 的值是长整型数 97 */
```

(2) 强制类型转换

强制类型转换将一个表达式强制转换成所需类型。在强制类型转换时,原来变量类型未发生变化。

例如,“(float)(7%4)”指的是将“7%4”的值强制转换成 float 型。

注意:表达式应该用括号括起来。如果写成“(int) x+y”,则只将 x 转换成整型,然后与 y 相加。

2.2 常见错误和难点分析

(1) 标识符。

标识符中大小字母代表不同的字符,如 SUM 和 sum 是两个不同的标识符。

定义标识符不能与 C 语言关键字及预定义标识符相冲突,如 main、define、sin、FILE。

(2) 变量和符号常量的定义。

对所有用到的变量、符号常量必须先定义后使用。定义时必须以字母或下画线开头,后跟字母、数字或下画线组成的字符串。

注意符号常量与变量的区别。符号常量在程序中用符号名来代替某一定义的常量,在程序中不能再赋值;而变量则是在程序中可以改变的量,根据需要随时可以再重新赋值。

(3) 变量的初始化。

对变量初始化只能单个进行,不能同时给多个变量初始化相同的值。

例如,以下是正确的写法:

```
int  x = 1, y = 1;
int  x, y; x = y = 1;
```

以下是错误的写法:

```
int  x = y = 1;
```

(4) 整数与整数相除结果为整数。

两个整数相除的结果为整数,在书写表达式时要特别小心,否则计算结果可能是错误的。

例如:代数式 $x = \frac{7}{11}y + 5$ 中, x, y 都是实数,假如改写成 C 语言表达式 $x = 7/11 * y + 5$,

不管 y 的值是什么, x 的值总是 5, 因为 $7/11$ 为 0, 这样的结果是错误的。

正确的写法如下:

```
x = 7.0/11 * y + 5
x = 7 * y/11 + 5
x = (float)7 * y/11 + 5
```