

高等教育 计算机基础课程系列教材

GAODENG JIAOYU
JISUANJI JICHU KECHENG XILIE JIAOCAI

C语言 程序设计

(第二版)

王彬华 易冬梅 编著



电子科技大学出版社

高等教育计算机基础课程系列教材

王彬华 易冬梅 编著

C语言 程序设计

(第二版)



电子科技大学出版社

图书在版编目(CIP)数据

C 语言程序设计/王彬华, 易冬梅编著. —成都: 电子科技大学出版社, 2004.4

ISBN 7-81094-429-0

I. C… II. ①王…②易… III. C 语言—程序设计
IV. TP312

中国版本图书馆 CIP 数据核字 (2004) 第 016469 号

内 容 简 介

C 语言功能丰富, 使用灵活, 可移植性好, 具有高级语言和低级语言的特点, 既可以用来编写系统软件, 又可用于编写应用软件, 是国内外广泛使用的计算机语言。C 语言程序设计是计算机相关专业人员必须掌握的基本功。针对 C 语言比较难学的情况, 本书在内容上做了精心安排, 降低难度, 突出重点, 用简洁的语言介绍 C 语言的基本语法, 同时使用大量的实例展现 C 语言在程序设计中的应用。

本书是高职高专学生学习 C 语言的理想教材, 也是一本想学 C 语言又苦于无从下手的编程爱好者的理想入门教材。

C 语言程序设计(第二版)

王彬华 易冬梅 编著

出 版: 电子科技大学出版社 (成都建设北路二段四号)

策划编辑: 谢应成 付金权

责任编辑: 周元勋

发 行: 新华书店经销

印 刷: 四川南方印务有限公司

开 本: 787×1092 1/16 印张 13.5 字数 328 千字

版 次: 2006 年 12 月第二版

印 次: 2006 年 12 月第一次印刷

书 号: ISBN 7-81094-429-0/TP·256

定 价: 22.00 元

■ 版权所有 侵权必究 ■

- ◆ 邮购本书请与本社发行科联系。电话: (028) 83201635 邮编: 610054
- ◆ 本书如有缺页、破损、装订错误, 请寄回印刷厂调换。电话: (0833) 7641415

前 言

程序设计是软件设计人员的基本功。C 语言由于其功能强、使用灵活、可移植性好、目标程序质量好而受到广泛的欢迎。目前,在高等学校,尤其是高职高专学校,各计算机类相关专业都把 C 语言程序设计开设为基础课程,而且越来越多的学校把 C 语言作为大一学生的入门必修课程。

如何做到让学生以更短的时间、更容易的方式迈过程序设计这道“门槛”,进入程序设计殿堂,是我们作为教师、编者要思考的问题。为了使《C 语言程序设计》一书更具有针对性(尤其是针对我国各类高职高专院校的学生),在编写该书时打破传统的以 C 语言语法为中心的编写模式,改为以应用为目标,在内容架构上面向应用、注意实用、精讲多练,引导学生通过“读—改—写”这一循序渐进的过程,最终让读者学会程序设计。

传统的 C 语言程序设计教材以语法为中心,精讲细讲 C 语言语法,让人感觉内容枯燥,教条死板,更为严重的是,学生学完《C 语言》课后普遍不会设计程序。为此,本教材设计以学生为中心,强调换位思考,从一个刚刚勉强学完高中数理化的大专一年级学生的角度思考问题,组织内容,设计案例和实训,目的是要让学生不仅掌握 C 语言基本的语法,同时让学生养成程序思维方式,并逐步学会分析问题、解决问题,并使用 C 语言设计程序。

本书作者长期担任 C 语言教学的教师,为了使本书更贴近教学实际,本书在设计时避免面面俱到,对内容进行了具有针对性的优化组合,突出 C 语言的基本的、核心的内容,包括: C 语言的数据类型、运算符与表达式、数据的输入输出、程序的三种结构(顺序结构、选择结构、循环结构)、数组、函数、指针、结构型与联合型、文件等模块。

全书分为十章,为了方便学生自学和教师教学,每一章开始均列出本章的总体要求、相关知识点、学习重点和难点等内容,每一章在结尾时进行一简明扼要的小结,在每一章的后面附有习题。同时,本书还对核心内容还设计了实训任务,以方便教学。

本书如有不足之处,请读者指正。

编 者

2006 年 11 月

目 录

第1章 C语言与程序的算法 1

- 1.1 简单的C程序及特点 1
 - 1.1.1 几个简单的C程序 1
 - 1.1.2 C程序的特点 3
- 1.2 C程序的上机操作 5
 - 1.2.1 C程序的上机操作过程 5
 - 1.2.2 C程序的编译器 6
 - 1.2.3 Visual C++ 6.0 环境下
C程序的操作过程 6
- 1.3 程序的算法 9
 - 1.3.1 算法的概念与特性 9
 - 1.3.2 算法描述方法 10
- 本章小结 13
- 习题一 13

第2章 数据类型、运算符与 表达式 15

- 2.1 标识符 15
- 2.2 C语言的数据类型 16
- 2.3 常量 17
 - 2.3.1 整型、实型及符号常量 .. 17
 - 2.3.2 字符型常量 19
- 2.4 变量 20
 - 2.4.1 变量的概念 20
 - 2.4.2 变量的定义 21
- 2.5 各类数值型数据之间的混合
运算 21
- 2.6 运算符与表达式 24
 - 2.6.1 算术运算符与算术表达式
..... 24
 - 2.6.2 赋值运算符与赋值表达式
..... 27

2.6.3 逗号运算符与逗号表达式 29

- 本章小结 30
- 习题二 30

第3章 数据的输入输出与顺序 程序设计 32

- 3.1 头文件与库函数 32
 - 3.1.1 头文件 32
 - 3.1.2 标准库函数 33
- 3.2 printf 与 scanf 34
 - 3.2.1 printf 函数 34
 - 3.2.2 scanf 函数 40
- 3.3 putchar 与 getchar 43
 - 3.3.1 字符输出函数 putchar()
..... 43
 - 3.3.2 字符输入函数 getchar()
..... 44
- 本章小结 45
- 习题三 45

第4章 选择结构程序设计 48

- 4.1 程序结构 49
 - 4.1.1 程序的三种基本结构 49
 - 4.1.2 复合语句 50
- 4.2 关系运算与关系表达式 50
- 4.3 逻辑运算与逻辑表达式 51
- 4.4 if 条件分支语句 53
 - 4.4.1 if 流程 53
 - 4.4.2 if else 流程 55
 - 4.4.3 else if 流程 57
 - 4.4.4 if 语句嵌套 60

4.5 switch 多路开关语句	61	7.1.1 函数的定义	111
本章小结	68	7.1.2 函数的参数	113
习题四	68	7.1.3 函数的返回值	114
 第5章 循环控制	 69	7.1.4 函数的调用	116
5.1 循环概述	69	7.2 函数的嵌套调用与递归调用	118
5.2 while 语句	70	7.2.1 函数的嵌套调用	118
5.3 do-while 语句	72	7.2.2 函数的递归调用	119
5.4 for 语句	76	7.3 数组作为函数参数	122
5.5 循环结构的嵌套	80	7.3.1 数组元素作函数的实参	122
5.6 break 语句和 continue 语句	83	7.3.2 数组名作函数参数	123
5.6.1 break 语句	83	7.4 变量的分类	126
5.6.2 continue 语句	84	7.4.1 变量的作用范围	126
本章小结	85	7.4.2 变量的存储方式	129
习题五	86	7.5 内部函数和外部函数	132
 第6章 数组	 89	7.5.1 内部函数	132
6.1 一维数组	89	7.5.2 外部函数	132
6.1.1 一维数组的定义	89	7.5.3 程序、文件、函数与变量 的关系	133
6.1.2 一维数组的引用	90	本章小结	134
6.1.3 一维数组的初始化	91	习题七	144
6.1.4 一维数组的应用实例	91	 第8章 指针、结构型与联合型	 137
6.2 二维数组	95	8.1 指针	137
6.2.1 二维数组的定义	95	8.1.1 指针的概念	137
6.2.2 二维数组的引用	96	8.1.2 指针变量的定义	138
6.2.3 二维数组的初始化	97	8.1.3 指针变量的赋值	139
6.3 字符数组与字符串	99	8.1.4 指针变量的运算	139
6.3.1 字符数组	99	8.1.5 指针和数组	143
6.3.2 字符串	101	8.2 枚举型、结构型	147
6.3.3 字符数组的应用实例	107	8.2.1 枚举型	147
本章小结	109	8.2.2 结构型	150
习题六	109	8.3 联合型	157
 第7章 函数	 111	8.3.1 联合型的概念	157
7.1 函数的定义与调用工作原理	111	8.3.2 联合型的定义	158
		8.3.3 联合型变量的说明	158
		8.3.4 联合型变量的赋值和使用	

.....	159	10.4.2	字符串读写函数 fgets 和 fputs.....	186
8.4 定义类型 — typedef	161	10.4.3	数据块读写函数 fread 和 fwrite.....	188
本章小结.....	162	10.4.4	格式化读写函数 fscanf 和 fprintf.....	190
习题八.....	163	10.5	文件的随机读写	191
第9章 预处理命令.....	166	10.5.1	文件定位 (rewind 函数 和 fseek函数)	191
9.1 概述.....	166	10.5.2	文件的随机读写	192
9.2 宏定义.....	166	10.6	文件检测函数.....	193
9.2.1 无参宏定义.....	167	10.6.1	文件结束检测函数.....	193
9.2.2 带参宏定义.....	170	10.6.2	读写文件出错检测函数	193
9.3 条件编译	172	10.6.3	文件出错标志和文件结 束标志置0 函数	194
本章小结.....	175	本章小结.....		194
习题九.....	175	习题十.....		194
第10章 文件.....	177	实验部份.....	195	
10.1 C文件概述	177	上机实验 一	195	
10.1.1 普通文件与设备文件..	177	上机实验 二	196	
10.1.2 ASCII 码文件与二进 制文件.....	178	上机实验 三	196	
10.2 文件指针	178	上机实验 四	197	
10.3 文件的打开与关闭.....	179	上机实验 五	198	
10.3.1 文件的打开 (fopen 函数)	179	上机实验 六	199	
10.3.2 文件关闭函数 (fclose 函数)	181	附录: C语言函数库.....	201	
10.4 文件的读写	181			
10.4.1 字符读写函数 fgetc 和 fputc.....	181			

第一章 C 语言与程序的算法

总体要求:

- 了解 C 程序的一般结构及特点
- 了解 C 程序的编辑、调试与运行等上机操作过程
- 理解算法的概念及其特性, 了解算法的描述方法

相关知识:

- 了解计算机软件、计算机语言及分类的知识
- 熟悉 Windows 系统基础知识及操作

学习重点:

- C 程序的结构、特点以及算法的概念

C 语言是 20 世纪 70 年代初出现的、国际上广泛流行的计算机高级语言, 它既可以用来编写底层系统软件, 也可以用来开发应用软件。目前, C 语言在我国已经成为学习程序设计的入门语言。本章将首先向大家展示几个简单的 C 程序, 并介绍 C 程序的特点, 然后在 Visual C++ 6.0 中演示 C 程序的编辑、编译、调试和运行的操作过程, 最后介绍程序算法的概念及其描述方法。

1.1 简单的 C 程序及特点

1.1.1 几个简单的 C 程序。

【实例 1-1】显示 Hello World

```
#include "stdio.h"

main()
{
    printf("Hello, World!\n");
}
```

本程序的作用是输出以下一行信息:

Hello, World!

在该程序中, #include 用来引用 C 语言的库函数文件。stdio.h 是标准输入输出库函数文件。main 表示主函数的名字。每一个 C 程序都必须有一个 main() 函数。函数体由若干行语句组成, 由大括弧 {} 括起来。函数体中的每一条语句以分号结尾。本例的函数体中只包含一条语句。Printf() 是 C 语言中的输出函数, 将圆括号 () 内的字符串显示输出, 字符串可

以是英文,也可以是中文。“\n”表示换行符,即在输出“Hello,World!”后自动回车换行。

【实例 1-2】计算并显示购物的应付款。

```
#include "stdio.h"

main()
{
    int a,b,money;    /*定义了三个变量名*/
    a=1219; b=23;     /*设单价 a=1219 元/台, 数量 b=23 台*/
    money=a*b;
    printf("应付款是%d 元\n",money);
}
```

本程序的作用是计算两个整数 a 和 b 的乘积,并显示以下一行结果信息:

应付款是28037元

在该程序中,主函数体的第一行中的 int 表示定义整数型变量。/*……*/表示注释,增加注释的目的是为了方便人阅读或修改程序,程序被编译时它将被忽略,在运行时不起作用。注释可以添加在程序中的任何位置。第三行“money=a*b”表示先将变量 a 和变量 b 的值相乘,再将乘积保存到变量 money 中。第四行中“%d”是输入输出的“格式字符串”,表示以十进制整数形式输出 money 的值。

【实例 1-3】显示使用字符拼成的图案。

```
#include "stdio.h"

main()
{
    void stars(void);    /*声明函数 stars*/
    stars();             /*调用函数 stars*/
    printf("* 你好,我是小罗 *\n");
    stars();             /*调用函数 stars*/
}

void stars()            /*定义函数 stars*/
{
    printf("*****\n");
}
```

本程序的作用是在输出以下三行字符组成的图案:

```
*****
* 你好,我是小罗 *
*****
```

该程序由两个函数组成,一个是主函数 main(),另一个是显示星号的函数 stars()。在该程序中,主函数体的第一行是声明函数 stars,stars 前的 void 表示不需要 stars()返回任何值给 main(),圆括号中的 void 表示不需要 main()传送任何值给 stars()。第二行和第四行表示重复调用函数 stars()。stars()函数的作用是显示一行由 18 个星号组成的图案,因此其函数体仅由一条 printf 语句组成。

【实例 1-4】求两个数的较大数。

```

#include "stdio.h"

main()
{
    int a,b,c;           /*定义三个整型变量 a、b、c*/
    int max(int,int);    /*声明函数 max*/
    printf("a="); scanf("%d",&a);    /*输入变量 a 的值*/
    printf("b="); scanf("%d",&b);    /*输入变量 b 的值*/
    c=max(a,b);          /*调用函数 max 并将结果返回给变量 c*/
    printf("max=%d\n",c); /*显示变量 c 的值*/
}

int max(int x,int y)    /*定义函数 max, x 和 y 是整型的形式参数*/
{
    int z;
    if(x>y) z=x;        /*if_else 表示汉语逻辑: 如果……, 否则……*/
    else z=y;
    return(z);          /*返回比较结果 z*/
}

```

本程序的作用是：首先输入两个任意整数，然后显示较大的那个数。本程序由两个函数组成：一个是主函数 main()，另一个是求较大数的函数 max()。在本程序中，max()函数的功能是：首先求出 x 和 y 中的较大者，并存放到变量 z 中，然后通过 return 语句将 z 的值返回给调用它的函数 main()。

main()函数体的第一行是定义三个整型变量 a、b、c，第二行是声明函数 max()，第三行和第四行提示用户运行程序时输入变量 a 和 b 的值，第五行调用函数 max()并返回较大数给变量 c，第六行显示变量 c 的值。其中的 scanf()是 C 语言中的输入函数，其后面的圆括号包括了两个东西：一个是"%d"，表示输入给变量 a 或 b 的值是十进制整数，另一个是&a 或&b，其中"&"表示“取地址”，不可缺少。scanf 语句执行的结果是将十进制整数的数值输入到变量 a 或 b 的地址所标志的内存单元中。main()函数在调用函数 max()时，首先将实际参数 a 和 b 的值分别传送给 max()函数中的形式参数 x 和 y，执行函数 max()后，即可获得 a 和 b 的较大数。

本程序的运行效果如下：

```

a=5
b=8
max=8

```

1.1.2 C 程序的特点

通过上述例子，我们可以看到，C 程序的结构和书写形式具有以下特点：

1. 函数是 C 程序的基本构成单位

一个 C 程序由一个主函数 `main()` 和若干个其他函数组成。C 程序中的主函数有且只能有一个, 其名字必须是 `main`, 其他函数可以有多个, 也可以没有。函数之间可以相互调用, 但主函数不能被其它函数调用。被调用的函数可以是系统提供的库函数 (例如 `printf()` 函数和 `scanf()` 函数), 也可以是用户自己定义的函数。用户自己定义的函数由用户自己命名 (须符合标识符的有关规定, 详见第二章标识符的定义)。每一个函数名后紧跟一对圆括号, 不能省略, 圆括号中可以带若干个参数, 也可以没有参数, 例如在实例 1-3 中的函数 `stars()`。

2. C 程序中的函数的组成

C 程序中一个函数由两部分组成: 函数的头部和函数体。

(1) 函数的头部

函数的头部即函数的第一行, 包括函数返回值类型、函数名、函数形参名及函数形参类型的说明。一个函数的形参可以没有, 也可以有多个。当一个函数带多个形参时, 形参之间用逗号隔开。例如, 在实例 1-4 的 “`int max(int x, int y)`” 中 `max` 是函数名, `x` 和 `y` 是函数的形参名, `int` 表示函数返回值和形参的类型都是整型。

(2) 函数体

函数体使用一对大括号 `{ }` 括起来, 通常由声明部分和执行部分组成。函数的声明部分用来定义即将使用的变量名或声明即将调用的函数, 例如在实例 1-4 中的语句 “`int a, b, c;`” 和 “`int max(int, int);`”, 前者表示定义三个整型变量 `a`、`b`、`c`, 后者表示声明函数 `max`。函数体的执行部分由若干个语句组成。一个函数体可以没有声明部分, 也可以没有执行部分, 甚至不包括任何语句, 例如:

```
empty()
```

```
{ }
```

它是一个空函数。

3. C 程序的执行

一个 C 程序总是从主函数 `main()` 开始执行, 而不论它处在整个程序的什么位置。因此, `main` 函数可以放在所有其他函数之前, 也可以放在所有其他函数之后。

4. C 程序的语句

C 程序中的每个语句必须以分号结尾。在书写时, 源程序的一行可以书写几条语句, 一条语句也可以分写在多行上。

5. C 程序的输入与输出操作

C 语言本身没有输入输出语句。输入输出操作是调用头文件 `stdio.h` 中的库函数 `scanf()` 和 `printf()` 来完成的, 因此在程序的最开始需要使用 `include` 来引用头文件 `stdio.h`。函数 `scanf()` 表示将数据输入到变量名所标识的内存单元, 例如在实例 1-4 中的 `scanf("%d", &a)`。函数 `printf()` 用来输出字符串数据, 如果需要输出变量的值, 则只需在字符串中用诸如 `%d` 的格式字符代替变量名并在右括号之前列出变量名即可, 例如在实例 1-2 中的 `printf("应付款是 %d 元\n", money)`。

6. C 程序的注释

在 C 程序中, 用户可以使用 `/*.....*/` 添加注释信息, 增加注释的目的是为了方便他人阅读或修改程序, 程序被编译时它将被忽略, 在运行时不起作用。注释可以添加在程序中

的任何位置。经验表明适当地添加注释，对程序的重要部分进行说明，可大大增强程序的可读性。

1.2 C 程序的上机操作

从上一节所展示的例子可以看出，C 程序实际上是根据问题的要求并遵照 C 语言的某些规定书写出来的东西。作为程序设计的初学者，要知道良好的程序设计习惯是先在纸上写好程序、再上机运行，切忌盲目上机。

1.2.1 C 程序的上机操作过程

在编好一个 C 程序后，那如何上机运行呢？在纸上写好一个程序后，要经过“上机输入并编辑源程序→对源程序进行编译→与库函数进行连接→运行可执行的目标程序”这样几个步骤，操作过程如图 1-1 所示。其中，实线表示操作流程，虚线表示文件的输入和输出。`.cpp`、`.obj`、`.exe` 分别表示完成编辑、编译、连接后所输出的文件的扩展名。

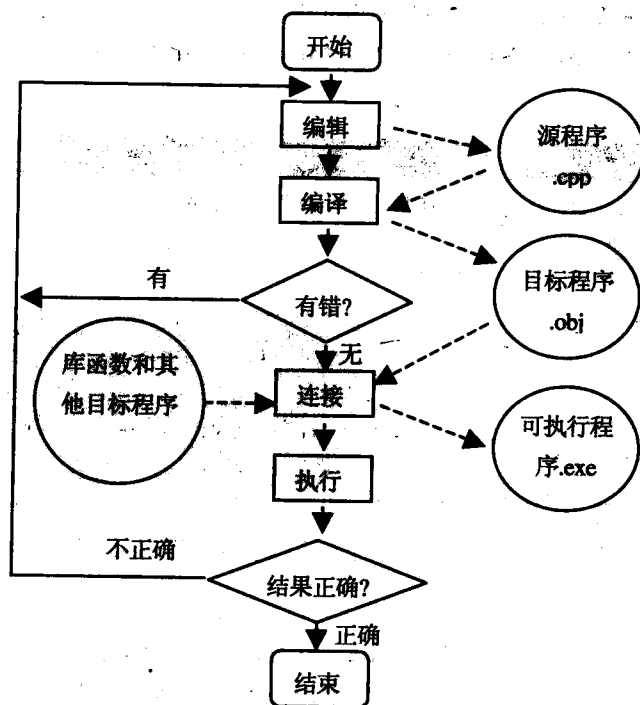


图 1-1 C 程序的上机操作过程

1.2.2 C 程序的编译器

C 语言源程序经过编辑保存后，将在磁盘上生成一个源程序文件，不同的编辑软件所生成的文件的扩展名往往是不相同的，但无论其扩展名是什么，源程序文件仍然是一个文本文件。正因为如此，我们可以使用任何一种文字处理软件完成源程序的编辑，但要注意保存为文本文件同时修改其扩展名，以符合你所使用的 C 语言编译器的要求。

目前，C 语言的编译器是比较多的，较常用的有：在 DOS 平台上使用的 Turbo C、在 Windows 平台上使用的 Borland C++ 和 Microsoft Visual C++ 以及在 Linux 平台上使用的 GCC 等等。本书以 Microsoft Visual C++ 6.0 为蓝本阐述 C 语言在程序设计上的应用。

1.2.3 Visual C++ 6.0 环境下 C 程序的操作过程

1. 编辑 C 语言源程序

Microsoft Visual C++ 6.0 是一个功能非常强大的集编辑、编译、连接、调试和运行程序为一体的软件开发平台。若要编辑 C 语言源程序，按如下操作步骤进行：

(1) 启动计算机，进入 Windows XP。

(2) 选择“开始→所有程序→Microsoft Visual Studio 6.0→Microsoft Visual C++ 6.0”菜单命令，以启动 Microsoft Visual C++ 6.0。Microsoft Visual C++ 启动后的操作界面如图 1-2 所示。

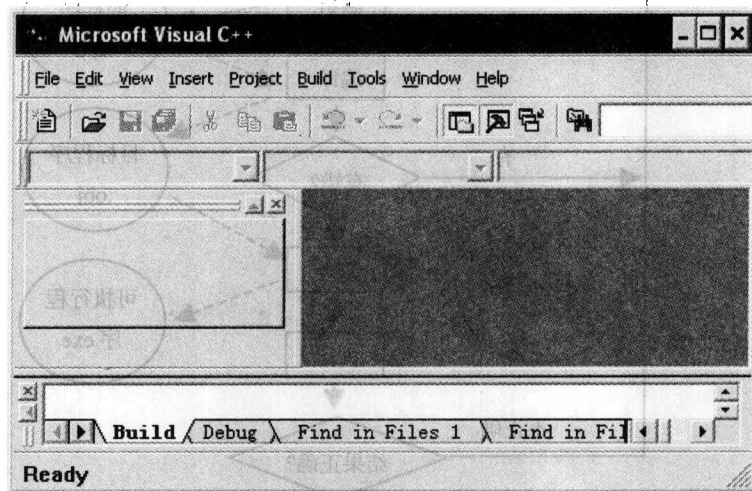


图 1-2 Microsoft Visual C++ 的主窗口

(3) 在 Microsoft Visual C++ 主窗口中选择“File→New”菜单命令，以弹出“New”对话框。

(4) 在“New”对话框选择“Files”选项卡，在列表框选择“C++ Source File”，然后在“File”文本框中输入想要创建的源程序文件名（如 myfile），并在“Location”文本框中输入存盘路径（如 d:\test），也可以通过单击右边的按钮 ，弹出“Choose Directory”对话框

框后选择磁盘路径。如图 1-3 所示。之后，单击“OK”按钮，返回 Microsoft Visual C++ 主窗口。

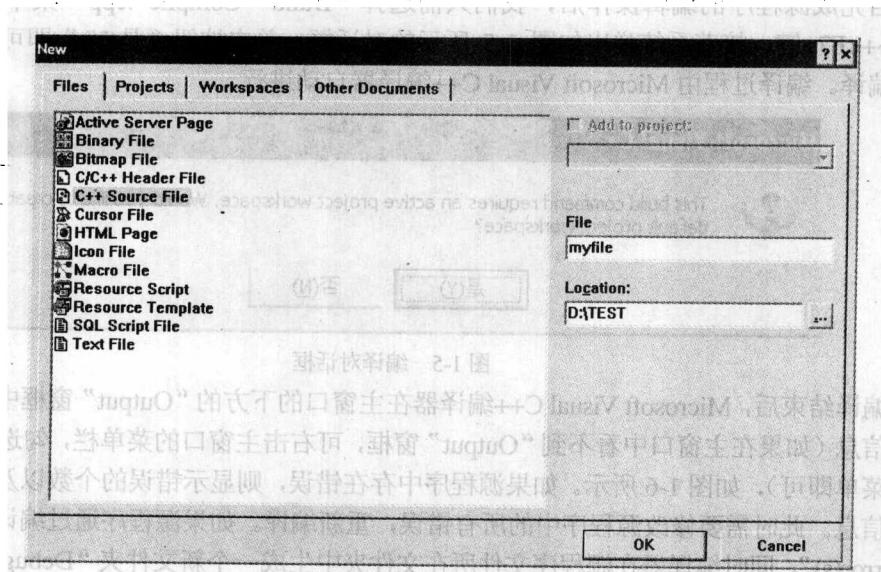
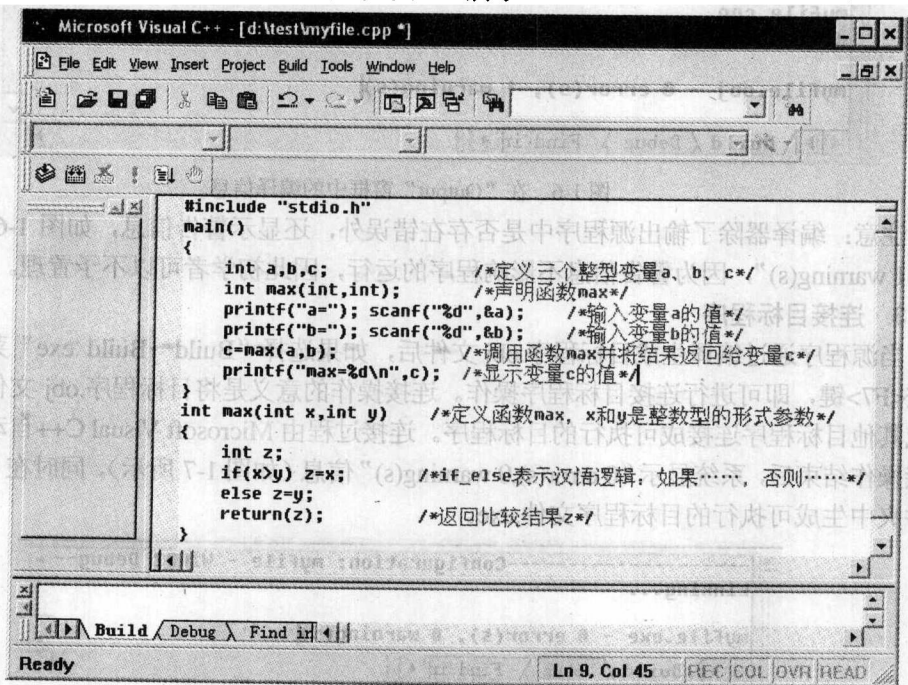


图 1-3 “New”对话框

(5) 在新出现的源程序文档窗口（如“d:\test\myfile.cpp”）中输入源程序代码。源程序文档的编辑操作与 Word 操作相同，如图 1-4 所示。



(6) 源程序文档编辑操作结束后，如果需要保存，可选择“File→Save”菜单命令即可。注意，在 Microsoft Visual C++ 中 C 语言源程序文件扩展名为.cpp。

2. 编译 C 语言源程序

当完成源程序的编辑操作后, 我们只需选择“Build→Compile .cpp”菜单命令或按<Ctrl>+<F7>键, 接着系统弹出如图 1-5 所示的对话框, 单击按钮“是(Y)”即可对源程序进行编译。编译过程由 Microsoft Visual C++ 编译器自动进行。

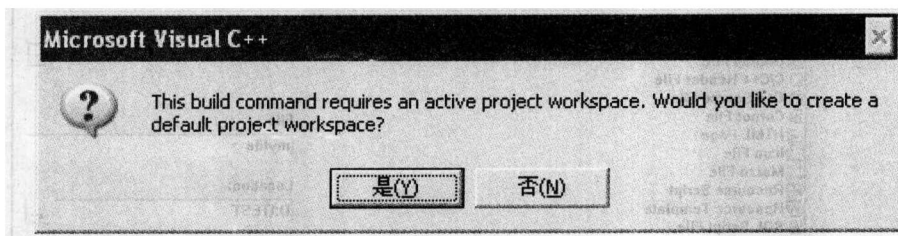


图 1-5 编译对话框

编译结束后, Microsoft Visual C++ 编译器在主窗口的下方的“Output”窗框中输出编译结果信息(如果在主窗口中看不到“Output”窗框, 可右击主窗口的菜单栏, 勾选“Output”快捷菜单即可), 如图 1-6 所示。如果源程序中存在错误, 则显示错误的个数以及错误原因分析信息。此时需要修改源程序中的所有错误, 重新编译。如果源程序通过编译, 则显示“0 error(s)”, 同时编译器在源程序文件所在文件夹中生成一个新文件夹“Debug”, 并将目标程序.obj 文件存入到该文件夹中。

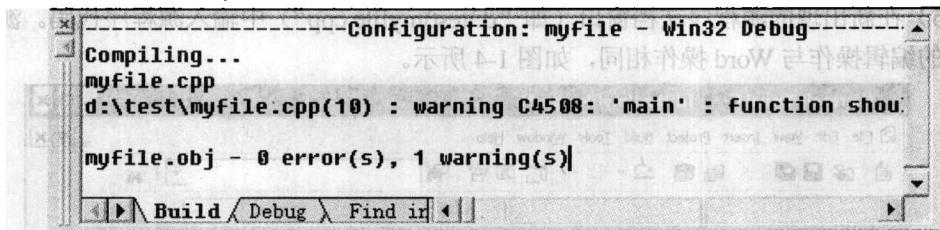


图 1-6 在“Output”窗框中的编译信息

注意: 编译器除了输出源程序中是否存在错误外, 还显示警告信息, 如图 1-6 中所示的“1 warning(s)”, 因为警告信息不影响程序的运行, 因此初学者可以不予理置。

3. 连接目标程序

当源程序通过编译生成目标程序.obj 文件后, 如果选择“Build→Build .exe”菜单命令或按<F7>键, 即可进行连接目标程序操作。连接操作的意义是将目标程序.obj 文件和库函数或其他目标程序连接成可执行的目标程序。连接过程由 Microsoft Visual C++ 自动完成。连接操作结束后, 系统显示“0 error(s), 0 warning(s)”信息(如图 1-7 所示), 同时在“Debug”文件夹中生成可执行的目标程序文件.exe。

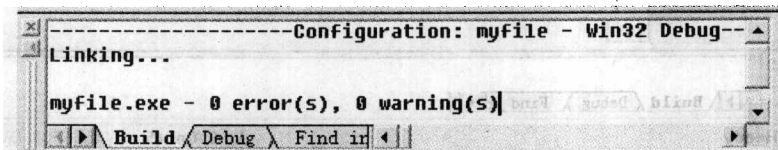


图 1-7 在“Output”窗框中的连接信息

4. 执行目标程序

当源程序经过编译和连接操作最终生成可执行的目标程序后, 在 Microsoft Visual C++

主窗口选择“Build→Execute .exe”菜单命令或按<Ctrl>+<F5>键,即可执行目标程序。

1.3 程序的算法

1.3.1 算法的概念与特性

1. 算法的概念

什么是程序?较权威的描述是:所谓程序就是一组计算机能识别和执行的指令(其中指令表示计算机能执行的特定操作)。对于初学者来说,这个描述往往让人一头雾水,不得要领。实际上,程序就是使用某一种计算机语言,根据问题的要求而编写出来的文档,如实例 1-4 所示。

编写程序的目的是:对问题中的数据进行分析或计算,最终得到我们想要的结果。对于一个复杂的问题,往往从两个方面入手解决,一是明确要解决的问题是什么,它隐含了哪些数据,二是针对这些复杂的数据该使用何种方法完成计算(即算法)。因此,一个程序也应该包括两方面的内容:一方面对数据进行描述,指定数据的类型和数据之间的关系(即数据结构),如在【实例 1-4】中的“int a,b,b;”;另一方面对数据的操作进行描述,根据操作步骤书写出能实现操作的语句序列,如在【实例 1-3】中的“a=1219; b=23; money=a*b; printf(“应付款是%d 元\n”,money);”。

可见,作为程序设计人员,必须认真考虑和设计数据结构和操作步骤(即算法)。著名计算机科学家沃思(Nikiklaus Wirth)提出一个公式:

数据结构 + 算法 = 程序

因此,对于人来说算法是为了解决一个问题而采取的方法,而对于计算机来说算法是具体的实实在在的操作序列。C 程序就是根据 C 语言的语法规则而编写的语句序列。语句的先后序列就体现了 C 程序的算法。

2. 算法的特性

一个算法应该具有以下特性:

(1) 有穷性

一个算法包含有限的操作步骤,而不能是无限的,否则意味着执行程序时永远不会停止。

(2) 确定性

算法中的每一步骤操作应当是确定的,而不应当是含糊的、模棱两可的。如果算法中具有不确定的操作描述,则算法无法通过编译,也就无法得到目标程序。例如,在【实例 1-3】的“a=1219; b=23; money=a*b;”中,如果去掉前面的“a=1219; b=23;”,则操作“money=a*b;”就是不确定的。

(3) 有零个或多个输入

如果运算所依赖的对象在程序设计时无法确定,则需要在程序执行时指定。这就意味着在执行程序时需要输入数据。一个算法可以没有输入,也可以有多个输入(例如在【实

例 1-4】中的 “scanf(“%d”,&a);” 和 “scanf(“%d”,&b);” 语句就是两个输入，它们负责将用户键入的数据存到变量 a 和 b 中。当然，算法的输入操作不一定就是敲键盘，例如在【实例 1-4】中的 “c=max(a,b);” 语句就包括了两个输入，它们负责将变量 a 和 b 的值分别输入到函数 “int max(int x,int y)” 中的变量 x 和 y。

(4) 有一个或多个输出

设计程序的目的是要得到想要的计算结果。而计算结果只有经过输出操作之后才有意义。当然算法的输出不一定就是计算机的显示或打印输出，例如在【实例 1-4】中函数 max() 的 “return z;” 语句就是一个输出，它负责将计算结果返回给 main() 函数。

(5) 有效性

有效性是指算法中的每一个操作步骤都应当有效地执行，并得到确定的结果。例如，若设 $x=1$, $a=0$ ，则 x/a 是不能有效执行的。

1.3.2 算法描述方法

算法的描述方法有，用自然语法表示算法、用流程图表示算法、用 N-S 盒图表示算法、用伪代码表示算法、用计算机语言表示算法等。下面重点介绍如何用自然语法、流程图和计算机语言描述算法。对于 N-S 盒图和伪代码表示算法问题，请参考相关书籍。

1. 用自然语言表示算法

自然语言就是人们日常使用的语言，可以是汉语、英语或其他语言。用自然语言表示算法，则算法通俗易懂，但文字冗长，容易出现歧义。自然语言所描述的内容往往不太严格，要根据上下文才能判断其正确含义。在算法分析的初始阶段使用自然语言描述算法是恰当的，尤其对于初学者来说，这是学习程序设计时迈出的第一步。

【实例 1-5】用自然语言描述 $1+2+3+\cdots+1000$ 的算法。

【分析】对于此问题，人工计算时有两种方法。第一种方法是先进行 $1+2$ ，再加 3，再加 4，一直加到 1000。这种方法称为逐个相加法。第二种方法是进行 $(1+1000)+(2+999)+\cdots+(500+501)=500*1001=500500$ 。逐个相加法虽然在人工计算时比较慢，但更具有规律性，因此适合计算机进行。在使用逐个相加法进行计算时，如果引入两个变量，一个变量代表要逐个相加的数，另一个变量代表每一次相加的结果。刚开始设前者的值为 1，后者的值为 0。在第一次将两个变量的值相加时，实际进行 0 和 1 相加，得 1。然后，将结果 1 存入到后一个变量中，再将前一个变量的值增加 1，进行第二次相加。在第二次将两个变量的值相加时，实际上进行 1 和 2 相加，得 3。然后将结果 3 存入到后一个变量中，再将前一个变量的值增加 1，再进行第三次相加……依次类推，直到前一个变量的值大于 1000 时，停止操作。最后，将后一个变量的值输出，所输出的结果就是希望得到的结果。而当计算停止的时候，我们一共重复了 1000 次相同的操作。

【算法】现在设前一个变量为 i，后一个变量 s，在进行每一次重复操作之后增加逻辑判断，判断 i 的值是否大于 1000，如果大于 1000，则停止计算，否则重复相同操作。相应算法描述如下：

S1: 把 1 存入变量 i 中；

S2: 把 0 存入变量 s 中；