

新一代信息技术系列教材

基于新信息技术的

C语言

主 编 马 庆 左向荣 左国才
主 审 符开耀 王 雷

程序设计基础教程



西安电子科技大学出版社
<http://www.xduph.com>

CONTROL

基于新信息技术的

C 语言程序设计基础教程

主 编 马 庆 左向荣 左国才

副主编 黄金水 魏红伟 曾 琴 黄利红

周海珍 苏秀芝 杨静宇 杨爱武

主 审 符开耀 王 雷

西安电子科技大学出版社

内 容 简 介

本书从高职高专学生的实际特点出发,以“实用、够用”为原则,采用通俗易懂的语言,通过具体的实例,深入浅出地介绍了 C 语言程序设计的基本概念和设计方法。全书共 11 章,介绍了 C 语言的数据类型、常量和变量、运算符和表达式,数据的输入/输出,顺序结构程序设计,选择结构程序设计,循环结构程序设计,数组的定义和使用方法,指针的定义和使用方法, C 语言中函数的定义、调用、参数传递以及变量的作用域和存储类型, C 语言的编译预处理指令, C 语言的结构体与共用体, C 语言文件的概念及操作等内容。本书中的全部例题、习题和上机实训内容均在 Visual C++6.0 环境下调试通过,便于读者直接上机验证。

本书既可作为高职高专院校学生“C 语言程序设计”课程的教材,也可作为全国计算机等级考试二级 C 语言的培训或自学教材。

图书在版编目(CIP)数据

基于新信息技术的 C 语言程序设计基础教程/马庆,左向荣,左国才主编. —西安:西安电子科技大学出版社,2018.9

ISBN 978-7-5606-5104-0

I. ① 基… II. ① 马… ② 左… ③ 左… III. ① C 语言—程序设计 IV. ① TP312.8

中国版本图书馆 CIP 数据核字(2018)第 221337 号

策划编辑 杨丕勇

责任编辑 买永莲

出版发行 西安电子科技大学出版社(西安市太白南路 2 号)

电 话 (029)88242885 88201467 邮 编 710071

网 址 www.xduph.com 电子邮箱 xdupfb001@163.com

经 销 新华书店

印刷单位 北京虎彩文化传播有限公司

版 次 2018 年 9 月第 1 版 2018 年 9 月第 1 次印刷

开 本 787 毫米×1092 毫米 1/16 印 张 12.25

字 数 286 千字

印 数 1~1000 册

定 价 30.00 元

ISBN 978-7-5606-5104-0 / TP

XDUP 5406001-1

如有印装问题可调换

前 言

C 语言是国内外广泛使用的一种计算机程序设计语言，其功能强大，使用方便、灵活，可移植性好，既具有高级语言的优点，又具有低级语言的许多特点，因此成为编制系统软件和应用软件的首选语言。

“C 语言程序设计”是我国很多高校都开设的一门重要的基础课程，在高职院校计算机专业的课程体系中尤为重要，它是学习其他程序设计语言及专业课程的基础。在本书的编写过程中，我们针对高职高专学生的特点和培养目标，从高职高专学生的实际出发，以“实用、够用”为原则，并结合编者多年从事 C 语言教学的实际经验，对全书的内容作了精心的安排，用通俗易懂的语言和与实际密切相关的例题深入浅出地介绍 C 语言程序设计的基本概念和设计方法。

本书是参考了国家教育考试委员会考试中心编写的《全国计算机等级考试考试大纲》中二级考试大纲的“C 语言程序设计考试要求”，以及部分省市计算机应用知识和应用能力水平考试大纲对 C 语言部分的要求编写而成的。本书在体系结构上尽可能地将概念、知识点与具体实例结合起来，同时借助于“说明”和“注意”等提示内容，帮助学生准确理解相关教学内容。另外，每章后面都有一定数量的与所讲内容以及计算机二级等级考试相匹配的习题和上机实训，其中，习题可帮助学生加深对教学内容的理解和掌握，上机实训则可帮助学生提高 C 语言编程的实际动手能力。

在编写本书的过程中，我们得到了许多从事计算机教学工作的同事的帮助和大力支持，他们对本书提出了很多宝贵的意见，在此向他们表示衷心的感谢。

虽然我们在编写过程中做了大量工作，但是由于水平有限，书中难免存在不足和疏漏的地方，敬请广大读者不吝赐教。

编 者
2018 年 6 月

目 录

第1章 C语言概论.....	1	3.2.3 字符输入/输出函数.....	27
1.1 C语言程序介绍.....	1	3.3 应用举例.....	29
1.1.1 C语言的发展.....	1	本章小结.....	30
1.1.2 C语言的特点.....	1	实训.....	30
1.2 C语言程序的组成.....	2	第4章 选择结构程序设计.....	32
1.3 C语言程序的实现.....	4	4.1 一个选择结构程序实例.....	32
1.3.1 C语言程序的运行过程.....	4	4.2 关系运算与逻辑运算.....	33
1.3.2 开发环境介绍.....	5	4.2.1 关系运算符与关系表达式.....	33
本章小结.....	9	4.2.2 逻辑运算符与逻辑表达式.....	34
实训.....	9	4.3 由if语句构成的选择结构.....	35
第2章 基本的数据类型与运算.....	10	4.3.1 if语句.....	35
2.1 一个简单的C程序设计实例.....	10	4.3.2 if-else语句.....	37
2.2 基本数据类型.....	11	4.3.3 if-else-if形式.....	38
2.2.1 整数类型.....	11	4.3.4 if的嵌套.....	39
2.2.2 实数类型.....	12	4.3.5 条件表达式.....	40
2.2.3 字符类型.....	12	4.4 switch语句和break语句.....	41
2.3 常量和变量.....	12	4.5 应用举例.....	43
2.3.1 常量.....	12	本章小结.....	45
2.3.2 变量.....	14	实训.....	46
2.4 运算符和表达式.....	15	第5章 循环结构程序设计.....	49
2.4.1 算术运算符与算术表达式.....	15	5.1 一个循环结构程序实例.....	49
2.4.2 赋值运算符与赋值表达式.....	16	5.2 while语句.....	50
2.4.3 数据类型转换.....	17	5.3 do-while语句.....	52
2.4.4 几个特殊的运算符.....	18	5.4 for语句.....	54
本章小结.....	19	5.5 多重循环.....	57
实训.....	20	5.6 break语句和continue语句.....	60
第3章 顺序结构程序设计.....	22	5.6.1 break语句.....	60
3.1 一个顺序结构程序实例.....	22	5.6.2 continue语句.....	60
3.2 数据的输入/输出.....	23	5.7 应用举例.....	61
3.2.1 输出函数printf.....	23	本章小结.....	64
3.2.2 输入函数scanf.....	26	实训.....	64

第6章 数组	67	8.2.3 参数传递	108
6.1 数组的一般定义形式	67	8.2.4 函数的嵌套调用	114
6.2 数组的表示方法	68	8.3 变量的作用域	115
6.3 一维数组	69	8.3.1 局部变量	115
6.3.1 一维数组的定义	69	8.3.2 全局变量	119
6.3.2 一维数组的初始化	69	8.3.3 变量存储类型与模块化 程序设计	120
6.3.3 一维数组的引用	70	本章小结	121
6.3.4 应用举例	71	实训	122
6.4 二维数组	73	第9章 预处理指令	126
6.4.1 二维数组的定义	73	9.1 宏定义	126
6.4.2 二维数组的初始化	74	9.1.1 字符串宏	126
6.4.3 二维数组的引用	74	9.1.2 带参数宏	127
6.4.4 应用举例	75	9.1.3 函数与宏的比较	128
6.5 字符串	79	9.2 文件包含	128
6.5.1 字符数组的一般操作方法	80	9.3 条件编译	128
6.5.2 字符串处理函数	82	本章小结	133
本章小结	85	实训	134
实训	85	第10章 结构体与共用体	135
第7章 指针	88	10.1 一个结构体的例子	135
7.1 指针的类型说明	88	10.2 结构体的定义与引用	136
7.2 指针变量的赋值	88	10.2.1 结构体类型的定义	136
7.3 数组指针	92	10.2.2 结构体变量的定义及初始化	137
7.3.1 数组指针的定义	92	10.2.3 结构体变量的使用	139
7.3.2 数组指针的运算	92	10.3 结构体数组与结构体指针	140
7.3.3 二维数组的指针	96	10.3.1 结构体数组	140
7.4 指针数组	98	10.3.2 结构体指针	142
7.5 字符串指针	99	10.4 链表	143
7.6 动态存储分配	99	10.4.1 链表概述	144
7.7 应用举例	101	10.4.2 链表的基本操作	146
本章小结	102	10.5 共用体	156
实训	103	10.5.1 共用体类型和共用体变量的 定义	157
第8章 函数	105	10.5.2 共用体成员变量的引用	157
8.1 函数的分类	105	10.6 枚举	158
8.2 函数的定义和调用	106	10.7 typedef 声明	159
8.2.1 函数的定义	106		
8.2.2 函数的调用和函数的返回	107		

本章小结.....	160	11.3 应用举例.....	167
实训.....	161	本章小结.....	174
第 11 章 文件	163	实训	175
11.1 文件概述	163	附录 模拟试题	176
11.1.1 文件的概念和类型.....	163	模拟试题一.....	176
11.1.2 文件指针.....	164	模拟试题二.....	180
11.2 文件的基本操作	164	模拟试题三.....	184
11.2.1 文件的打开和关闭.....	164	参考文献	188
11.2.2 文件的读/写	165		

第 1 章 C 语言概论

C 语言是一种计算机程序设计语言，既有高级语言的特点，又具有汇编语言的特点。它可以作为系统设计语言，用于编写工作系统应用程序，也可以作为应用程序设计语言，用于编写不依赖计算机硬件的应用程序。因此，它的应用范围十分广泛。

1.1 C 语言程序介绍

1.1.1 C 语言的发展

早期的操作系统等系统软件主要是用汇编语言编写的(包括 UNIX 操作系统在内)，但是汇编语言存在明显的缺点，即用其编写的程序的可读性和可移植性都比较差。为了提高程序的可读性和可移植性，可改用高级语言，但是一般的高级语言难以实现汇编语言的某些功能(汇编语言可以直接对硬件进行操作，例如对内存地址的操作、位操作等)。为此人们希望能找到一种既具有一般高级语言的特性，又具有低级语言底层操作能力的语言，即集它们的优点于一体的语言，于是 C 语言在 20 世纪 70 年代初应运而生了。1978 年由美国电话电报公司(AT&T)的贝尔实验室正式发表了 C 语言，同时由 B. W. Kernighan 和 D. M. Ritchie 合著出版了影响深远的《THE C PROGRAMMING LANGUAGE》一书，该书通常被简称为《K&R》，也有人称之为《K&R》标准。在《K&R》中并没有定义一个完整的标准 C 语言，因此，许多开发机构推出了自己的 C 语言版本，而这些版本之间的微小差别不时产生兼容性上的问题。针对这些问题，美国国家标准学会(American National Standard Institute, ANSI)在各种 C 语言版本的基础上制定了一个 C 语言标准，于 1983 年发表，通常称之为 ANSI C。

早期的 C 语言主要用于 UNIX 系统，其由于强大的功能和各方面的优点而逐渐为人们所认识。20 世纪 80 年代，C 语言开始进入其它操作系统，并很快在各类大、中、小和微型计算机上得到广泛的使用，成为当代最优秀的程序设计语言之一。

1.1.2 C 语言的特点

C 语言是一种通用、灵活、结构化、标准化、使用广泛的编程语言，能完成用户的各种任务，特别适合进行系统程序设计和对硬件进行操作的场合。C 语言本身不对程序员施加过多限制，是一种专业程序员优先选择的语言。

C 语言有如下主要特点：

(1) 简洁紧凑, 压缩了一切不必要的成分。ANSI C 只有 32 个关键字, 书写形式自由。

(2) 运算符丰富, 并将括号、赋值、强制类型转换、取变量地址等都以运算形式实现。ANSI C 提供了 34 种运算符, 灵活使用这些运算符可以实现其它高级语言难以实现的操作。C 语言的表达式简练、多样、灵活、实用, 加上分号可以构成语句。

(3) 数据类型丰富, 具有现代语言的各种数据类型, 用户还能扩充它, 实现各种复杂的数据结构, 完成各种问题的数据描述。尤其是 C 语言的指针类型, 非常有特色, 可指向各种数据, 完成数据的高效处理。C 语言不但能对数据作类型上的描述, 还提供了存储属性方面的考虑。

(4) 它是一种结构化的程序设计语言, 层次清晰, 便于按模块化方式组织程序, 易于调试和维护。C 程序由若干程序文件组成, 一个程序文件由若干函数构成。

(5) 可以直接访问物理地址, 进行位(bit)一级的操作, 能实现汇编语言的大部分功能。由于 C 语言实现了对硬件的编程操作, 因此 C 语言集高级语言和低级语言的功能于一体, 表现能力和处理能力极强, 有时也被称作中级语言。

(6) 提供了预处理机制, 有利于大型程序的编写和调试。

(7) 生成的目标代码质量很高, 故程序执行效率很高, 一般只比汇编程序生成的目标代码效率低 10%~20%。

(8) 用 C 语言编写的程序可移植性好(与汇编语言相比), 基本上不做修改就可以用于各种型号的计算机和各种操作系统。

(9) 语法限制不太严格, 程序员的设计自由度较大。例如, 对数组下标越界不做检查, 由程序员自己保证程序的正确性。一般的高级语言语法检查都比较严格, 能检查出几乎所有的语法错误, 而 C 语言允许程序员有较大的自由度, 因此放宽了语法检查。限制与灵活是一对矛盾, 限制严格, 就会失去灵活性; 而强调灵活, 就必然放松限制。一个不熟练的程序员, 编写一个正确的 C 语言程序可能会比编写一个其它高级语言程序难一些。也就是说, 对于使用 C 语言的人, 要求其对程序设计要更熟练一些。

1.2 C 语言程序的组成

下面通过几个简单的程序实例来说明 C 语言程序的组成。

【例 1.1】 在屏幕上输出一串字符 “Hello World!”。

程序代码:

```
#include <stdio.h>

void main()
{
    printf("Hello World! ");
}
```

运行结果:

Hello World!

分析:

(1) `main()`为主函数。C语言程序由函数构成,但有且只有一个主函数。

(2) C语言程序中必须至少有一对`{ }`,代表程序的开始、结束,其中的内容称为函数体。

(3) `printf()`为标准输出函数,用于将程序运行结果显示到输出设备(显示器)上。

(4) `#include <stdio.h>`为预处理命令,当程序中有输出函数或输入函数时必须有此行。

【例 1.2】 已知 $a = 10$, $b = 30$, 求两数之和 sum 。

程序代码:

```
#include <stdio.h>

void main()                /*主函数, 程序从此开始运行*/
{                            /*函数体开始*/
    int a, b, sum;          /*定义语句*/
    a = 10;                 /*赋值语句*/
    b = 30;                 /*赋值语句*/
    sum = a+b;              /*赋值语句*/
    printf("sum = %d\n", sum); /*输出语句*/
}
```

运行结果:

```
sum = 30
```

分析:

(1) “`int a, b, sum;`”语句说明 a 、 b 和 sum 为三个整型变量,可以通过赋值操作改变变量的值。

(2) “`a = 10;`”语句表示将整数 10 送到 a 的存储单元中。“`sum = a + b;`”语句表示先取 a 和 b 两个存储单元中的数据并在运算器中相加,然后将结果保存在 sum 变量单元中。

(3) “`printf("sum = %d\n", sum);`”语句表示该函数的参数包括两部分, sum 是要输出的数据, “`%d`”是数据输出的格式控制符字符串,控制符 “`%d`”的作用是按整数格式输出 sum 的值。“`sum=`”是输出数据的提示说明,原样输出。

(4) “`/*...*/`”表示注释,目的是增加程序的可读性。注释可以插入到程序中任何位置,对程序的执行没有任何影响,编译时将被过滤掉。

【例 1.3】 求最大数。

程序代码:

```
#include "stdio.h"

int max(int x,int y)
{
    int z;
    if(x > y) z = x; else z = y;
    return(z);
}
```

```
}  
void main()  
{  
    int a, b, c;  
    printf("输入两个整数 a b:");  
    scanf("%d %d", &a, &b);           /*标准输入函数*/  
    c = max(a,b);                     /*调用 max()函数*/  
    printf("%d %d 中的最大值为: %d\n", a, b, c); /*输出结果 */  
}
```

运行结果:

输入两个整数: a b: 10 17

10 17 中的最大值为: 17

分析:

(1) 本程序由两个函数组成, 即主函数 main()和自定义函数 max()。

(2) 程序从 main()函数开始执行; printf()函数表示输出显示一个字符串, 具有提示信息的作用; scanf()函数表示从键盘输入数据, 分别给变量 a 和变量 b 赋值, 使得变量 a 和 b 从键盘上获取值; 执行 “c = max(a, b);” 语句时程序转移到 max()函数, 遇到 “return” 语句则返回主函数继续执行。

通过以上实例可以看出 C 语言程序的基本组成:

(1) C 语言程序是由函数构成的, 函数是 C 程序的基本单位。

(2) 一个函数由函数头和函数体两部分组成, 函数头即函数的第一行, 函数体即函数头下面用大括弧{ }括起来的部分。

(3) 函数体由语句构成, 语句以分号(;)结束。

(4) 一个 C 程序可以由一个或多个函数组成, 但必须有一个且只能有一个 main()函数, 即主函数。一个 C 程序总是从 main()函数开始执行的, 而不论 main()函数在整个程序中的位置如何。

(5) 通常, 每行只写一条语句, 短语句可以一行写多条, 长语句则可以一条分成多行来写。

(6) 在程序中应尽量使用注释信息, 以增强程序的可读性。注释信息是用注释符标识的, 注释符以 “/*” 开头, 以 “*/” 结束, 其间的字符为注释信息。

1.3 C 语言程序的实现

1.3.1 C 语言程序的运行过程

由高级语言编写的程序称为源程序, 它不能被计算机直接识别和执行, 必须由语言处理程序将其翻译成由 0 和 1 构成的二进制指令代码。按照 C 语言规则编写的程序, 要想得到最终结果, 则必须经历以下几个步骤。其实现过程如图 1-1 所示。

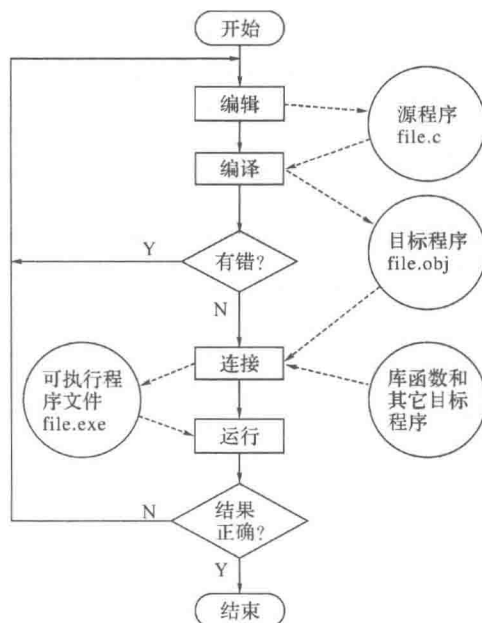


图 1-1 C 程序的实现过程

1. 编辑

编辑是指使用文本编辑软件输入和修改 C 语言源程序，最后以文本文件的形式存放在磁盘上，文件名由用户自己选定，扩展名一般为“.c”。编辑器可以是任何一种文本编辑软件，比如 TURBO C 和 VC++ 专用编辑系统，也可以是写字板、记事本等字处理软件。

2. 编译

编译是将 C 语言源程序翻译成二进制目标程序。编译是由编译程序来完成的。编译程序对源程序自动进行句法和语法检查，当发现错误时，就将错误的类型和程序中出错的位置显示出来，以帮助用户修改源程序；如果未发现句法和语法错误，就自动形成目标程序，其扩展名为“.obj”。

3. 连接

编译后的目标文件尽管是二进制代码文件，但计算机还不能直接执行该程序，必须使用 C 语言处理系统提供的连接程序，生成扩展名为“.exe”的可执行文件。程序中各函数间的调用结合是由连接程序完成的，系统提供的输出函数和用户定义的函数都要进行连接。如果连接过程中出现错误信息，则必须回到第一步修改源程序，重新开始编译和连接，直到生成可执行文件。

4. 运行

运行程序，并检查运行结果。如果是算法错误，则只能回到第一步修改源程序，再重新编译、连接和运行，直到得到正确的结果。

1.3.2 开发环境介绍

C 语言的标准已被大多数 C 和 C++ 的开发环境所兼容，因此用户可以使用多种工具开

发自己的 C 语言程序。下面以 Microsoft Visual C++ 6.0 为平台, 介绍 C 程序的实现过程。

VC++ 集成开发环境不仅支持 C++ 程序的编译和运行, 也支持 C 语言程序的编译和运行。通常 C++ 集成环境约定: 当源文件的扩展名为 “.c” 时, 则为 C 程序; 而当源文件的扩展名为 “.cpp” 时, 则为 C++ 程序。

(1) 启动 VC++, 进入图 1-2 所示的界面。

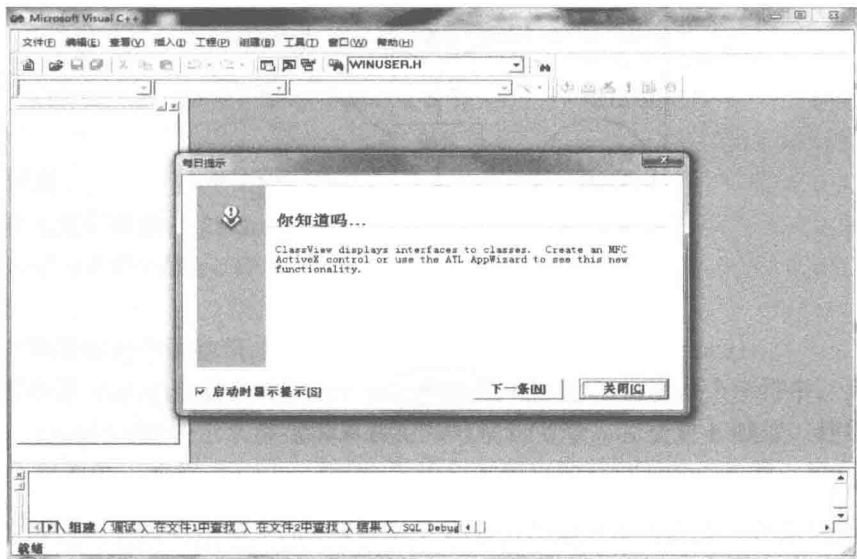


图 1-2 VC++ 6.0 启动界面

(2) 点击“文件”→“新建”菜单项, 如图 1-3 所示, 可新建一个工程。

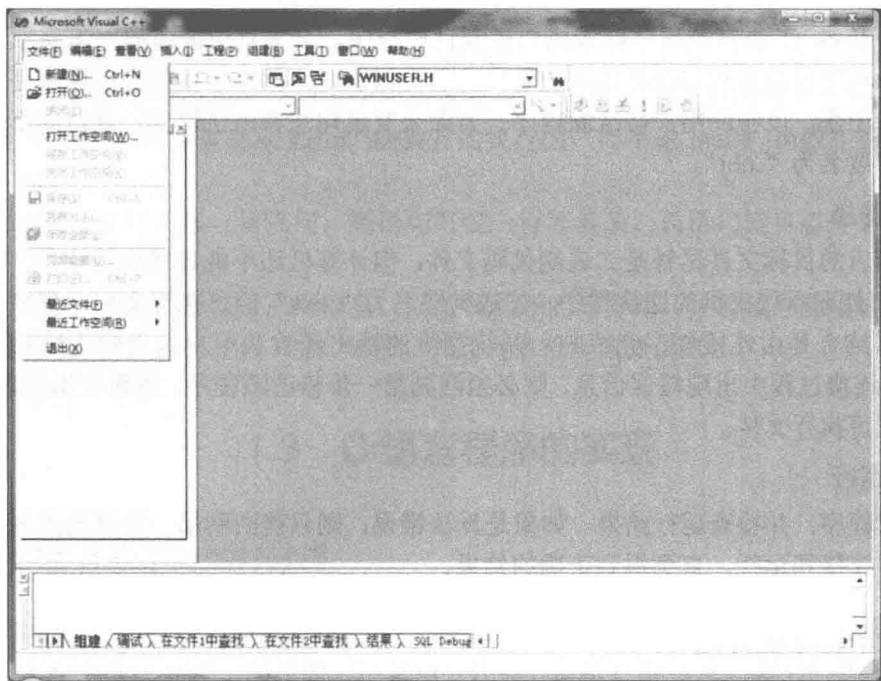


图 1-3 新建工程

(3) 在出现的如图 1-4 所示的界面中, 选择“Win32 Console Application”(控制台应用程序, 左边倒数第三个), 命名工程名称, 选择保存位置, 然后点击“确定”按钮, 进入下一步。

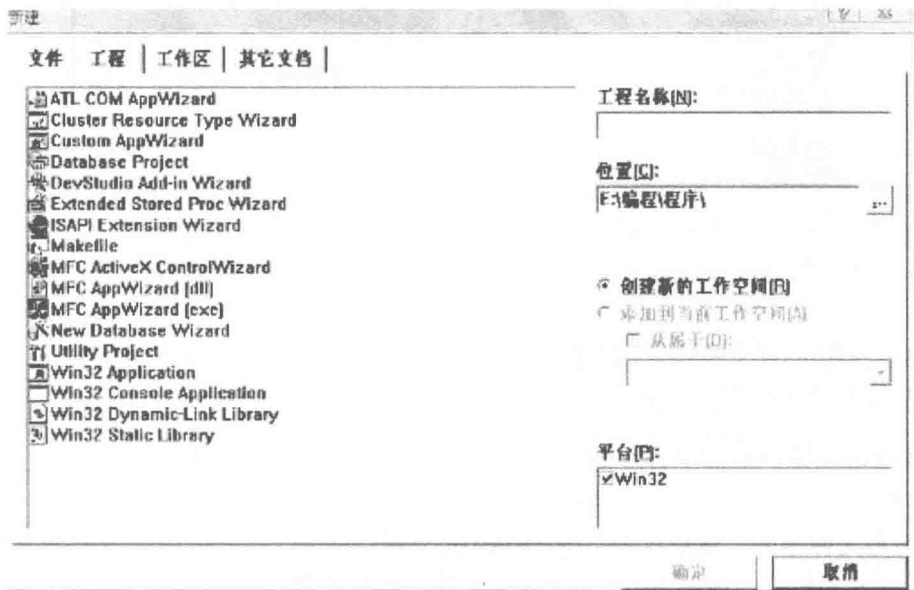


图 1-4 新建工程

(4) 在出现的如图 1-5 所示的界面中, 选择“一个空工程”项, 建立一个空工程。其他项可根据需要选择建立别的工程。然后点击“完成”按钮, 即显示所创建的工程的信息。

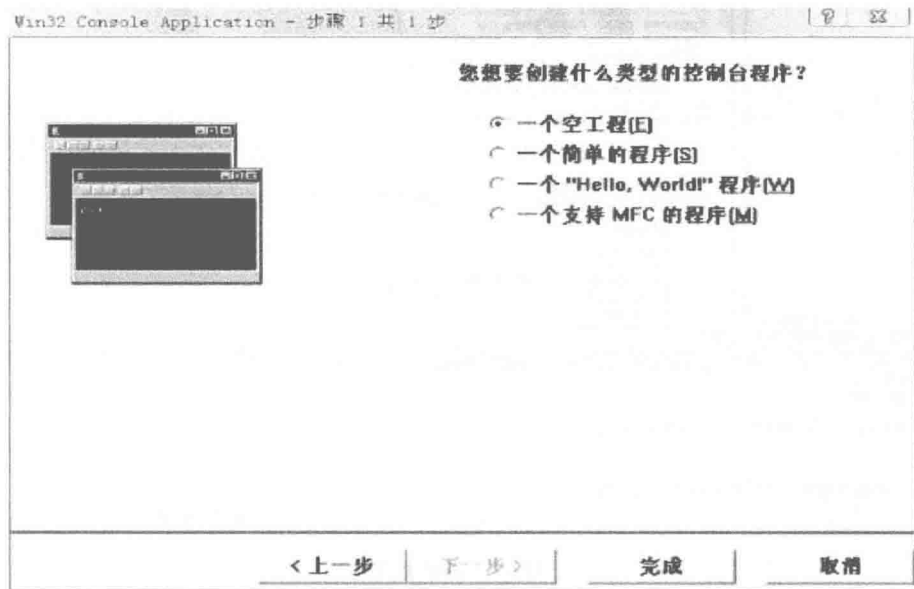


图 1-5 生成空工程

(5) 现在在有一个工程的条件下, 再建立一个源文件。点击“文件”→“新建”菜单

项(或按快捷键 Ctrl + N), 出现如图 1-6 所示的界面。

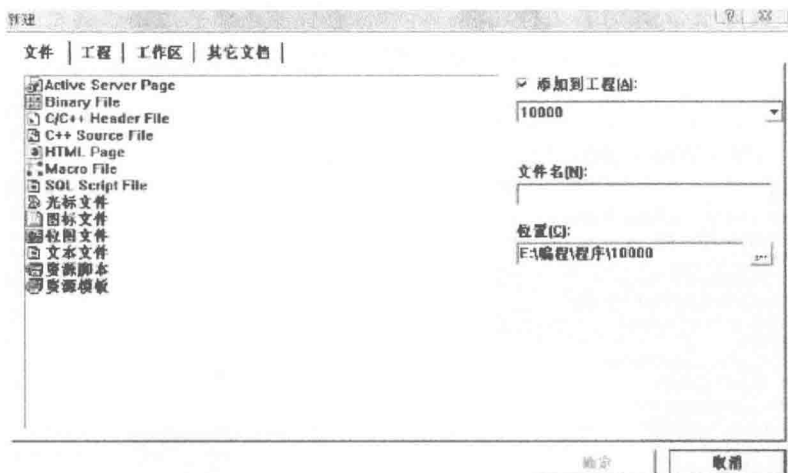


图 1-6 新建源文件

在图 1-6 所示的界面中建立源文件, 即选择“C++ Source File”项(一般都是建立这种文件)。如果要建立头文件, 可选择“C/C++ Header File”项, 这种文件适用在多文件工程中。然后为文件命名, 最后点击“确定”按钮。

(6) 进入编辑区, 如图 1-7 所示在主界面中可编写代码。



图 1-7 编辑程序

(7) 编译程序。选择编译工具栏中的编译按钮  进行编译, 编译信息显示在输出窗口中。如有错误, 则必须修改源程序再重新编译, 否则无法进行下一步。

(8) 连接程序。选择编译工具栏中的“构建”按钮  进行连接。如有错误, 则必须

修改源程序再重新编译和连接。

(9) 运行程序。选择编译工具栏中的“执行”按钮即可运行程序。

本章小结

本章主要介绍了以下内容：

1. C 语言的历史以及 C 语言的特点。
2. C 语言源程序的书写规范，如语句的结束符号、语句的书写习惯、注释的合理使用等。这些规范有的是强制执行的，即如果不这样做就会导致编译错误；另外有一些是经验总结，可使程序易于理解和调试。因此一开始就应养成自觉遵守书写规范的好习惯。

3. C 语言的组成以及在 VC++ 6.0 中的实现过程。

此外，本章还介绍了 C 语言的结构特征、主函数的作用及位置无关性(最先执行)、函数的分类及库函数、头文件及预处理、标识符的构成规则等基础知识。在等级考试中，这些内容经常出现在选择题中，尤其是标识符的构成规则、主函数的特性等。

实 训

1. 输入并运行例题中的程序，熟悉调试 C 程序的方法与步骤。
2. 编写一个 C 程序，熟悉其构成。
3. 参照例题，编写一个 C 程序，输出以下信息：

Hello, World!

4. 编写一个 C 程序，输入 a、b、c 三个数，输出其中最大者。试想，如果求四个数中的最大者，则程序该如何编写。

第 2 章 基本的数据类型与运算

本章将介绍用 C 语言编程必须掌握的一些基础知识,如最基本的数据类型、常量和变量、算术运算符及算术表达式、赋值运算符和赋值表达式及自增自减运算等。

2.1 一个简单的 C 程序设计实例

【例 2.1】 求矩形的面积。

程序代码:

```
#include <stdio.h>

void main()
{
    int a, b;
    float area;
    a = 2;
    b = 4;
    area = a*b;
    printf("%f\n", area);
}
```

运行结果:

8.000000

程序分析:

(1) 该程序中用到的数据有 a、b、area、2、4,对数据进行的运算有*(乘法运算)和=(赋值运算)。

(2) 计算机在运行程序时,要完成以下工作:

① 在内存中给矩形的两条边 a、b 和面积 area 开辟存储空间,存放它们的值。这里, a、b、area 被称为变量。

② 数据 2、4 与 a、b、area 不同,它们是在编写程序之初就给出了确定的值,在运算过程中不会改变,因此这样的数据叫做常量。

以上的叙述都涉及 C 语言中数据的处理操作。

在 C 语言中,数据是程序的必要组成部分,是程序处理的对象。现实中的数据是有类型差异的,如姓名由一串字符组成,年龄是数字符号组成的整数,而身高包含整数和小数两部分。C 语言中将不同类型的数据以不同的格式存储,占用内存单元的字节数也不同。