



# 沟通

## ——从C语言开始

叶子青 徐 慧 编著



人民邮电出版社  
POSTS & TELECOM PRESS

TP312  
1501

# 沟通

## ——从C语言开始

叶子青 徐 慧 编著

北方工业大学图书馆



00568379

 **人民邮电出版社**  
POSTS & TELECOM PRESS

## 图书在版编目 (CIP) 数据

沟通——从 C 语言开始 / 叶子青, 徐慧编著. —北京: 人民邮电出版社, 2005.1

ISBN 7-115-12913-4

I. 沟... II. ①叶...②徐... III. C 语言—程序设计 IV. TP312

中国版本图书馆 CIP 数据核字 (2004) 第 128866 号

## 内容提要

本书从学习 C 语言所要具备的最基础的知识讲起, 循序渐进地介绍了 C 语言的编程环境、数据类型和运算、流程控制及相关的语句、数组、函数、指针、结构体、共用体、枚举类型、位运算、输入和输入函数等各个方面的知识, 并在最后一章给出了 C 语言编程的建议和技巧。

本书体裁新颖, 实例程序丰富, 并配以大量的练习, 力求使没有编程基础的读者能够尽快掌握和应用 C 语言。本书可作为 C 语言初学者理想的入门读物。

### 沟通——从 C 语言开始

---

◆ 编 著 叶子青 徐 慧

责任编辑 汤 倩

◆ 人民邮电出版社出版发行 北京市崇文区夕照寺街 14 号

邮编 100061 电子函件 315@ptpress.com.cn

网址 <http://www.ptpress.com.cn>

读者热线 010-67132692

北京鸿佳印刷厂印刷

新华书店总店北京发行所经销

◆ 开本: 787×1092 1/16

印张: 20

字数: 482 千字

2005 年 1 月第 1 版

印数: 1-6 000 册

2005 年 1 月北京第 1 次印刷

ISBN 7-115-12913-4/TP · 4341

---

定价: 29.00 元

本书如有印装质量问题, 请与本社联系 电话: (010) 67129223

# 前言

在众多的计算机编程语言中，C 语言的重要地位是无可取代的。学好 C 语言成了很多高校理工科专业对学生的基本要求，也是很多 IT 人必须掌握的一门基本的编程语言，甚至很多初、高中生也加入了 C 语言的学习行列。

很多 C 语言的初学者一般不具有较深的计算机知识，因而他们渴望有一本能够浅显地介绍 C 语言的编程知识，并能方便上手的教程或参考书。阅读本书时，读者不需要有任何的编程经验，只要跟随书中的内容循序渐进地进行学习，便能够顺利地学会 C 语言，多花费一些时间和精力，还能够达到熟练使用 C 语言的目的，这也正是本书——“沟通，从 C 语言开始”策划的由来。

与同类的 C 语言入门书籍比较，本书的特色如下：

- ◇ 使用浅显但生动的语言，充分调动学习者的阅读和动手兴趣，使读者对于计算机语言的学习不再有恐惧感；
- ◇ 通过大量的实例以及比较和分析，帮助读者辨析 C 语言中容易混淆的内容；
- ◇ 每一章均配有大量习题，便于学习者及时巩固所学内容；
- ◇ 书中穿插介绍了作者多年来总结的 C 语言的编程经验和技巧，目的是使读者能够尽快养成编写程序的良好习惯。

本书主要由叶子青编写，参加写作的人员还有徐慧、沈炜、刘海英等。在写作过程中，我们力求精益求精，但书中难免存在考虑不周之处，热忱欢迎读者批评和指正。如果读者在阅读本书时遇到困难，可以发 E-mail 至 [tangqian@ptpress.com.cn](mailto:tangqian@ptpress.com.cn) 与我们联系。

编 者

2005 年 1 月

# 目 录

|                                 |    |
|---------------------------------|----|
| 第 0 章 C 语言基础                    | 1  |
| 0.1 第二天, 上帝创造了 C                | 2  |
| 0.2 应该准备什么                      | 3  |
| 0.3 热身运动                        | 3  |
| 0.4 留言板程序                       | 14 |
| 0.5 一个程序的产生到结束                  | 17 |
| 0.6 小结                          | 18 |
| 0.7 习题                          | 18 |
| 第 1 章 C 语言数据类型                  | 21 |
| 1.1 基础中的基础                      | 22 |
| 1.1.1 变量及其定义过程                  | 24 |
| 1.1.2 int、float (double) 和 char | 26 |
| 1.1.3 变量赋初值                     | 29 |
| 1.1.4 进制及换算                     | 30 |
| 1.1.5 我变我变我变我变                  | 35 |
| 1.1.6 坚定的常量                     | 37 |
| 1.2 运算符和表达式并不像看上去那么简单           | 41 |
| 1.2.1 +、-、*、/和%                 | 42 |
| 1.2.2 “++”和“--”                 | 43 |
| 1.2.3 多样化的赋值运算符及其表达式            | 45 |
| 1.2.4 逗号运算符和逗号表达式               | 46 |
| 1.2.5 孰大孰小                      | 47 |
| 1.2.6 真真假假                      | 48 |
| 1.2.7 言简意赅的“?:”表达式              | 50 |
| 1.2.8 变量的量具 sizeof              | 52 |
| 1.2.9 排一下顺序是绝对必要的               | 52 |
| 1.3 小结                          | 54 |

|                                       |     |
|---------------------------------------|-----|
| 1.4 习题                                | 55  |
| 第 2 章 C 语言的流程控制                       | 57  |
| 2.1 C 语言程序设计的结构及基本语句                  | 58  |
| 2.1.1 程序的 3 种基本结构                     | 58  |
| 2.1.2 C 语言的语句                         | 59  |
| 2.1.3 赋值语句                            | 61  |
| 2.1.4 数据输出语句                          | 61  |
| 2.1.5 数据输入语句                          | 66  |
| 2.2 岔路口的抉择                            | 69  |
| 2.2.1 if...else 和 else if             | 70  |
| 2.2.2 指哪打哪的 switch                    | 73  |
| 2.3 不可缺少的循环体                          | 76  |
| 2.3.1 一劳永逸的 while 语句和 for 语句          | 76  |
| 2.3.2 貌合神离的 while 和 do while 语句       | 81  |
| 2.3.3 break 还是 continue               | 82  |
| 2.3.4 千里之堤往往溃于蚁穴                      | 84  |
| 2.4 还差一点点                             | 85  |
| 2.4.1 分不开的 return 和函数                 | 86  |
| 2.4.2 自由主义的 goto                      | 86  |
| 2.5 小结                                | 87  |
| 2.6 习题                                | 89  |
| 第 3 章 C 语言的数组和字符串                     | 91  |
| 3.1 打开一个更广阔的视野——一维数组                  | 92  |
| 3.1.1 一维数组和字符串                        | 92  |
| 3.1.2 一维数组和字符串的运算                     | 101 |
| 3.2 向更高维进军——二维数组和 multidimensional 数组 | 110 |
| 3.2.1 二维数组的基本概念                       | 110 |
| 3.2.2 二维字符数组                          | 114 |
| 3.2.3 二维数组程序举例                        | 115 |
| 3.2.4 三维及三维以上数组                       | 121 |
| 3.3 小结                                | 121 |
| 3.4 习题                                | 121 |
| 第 4 章 C 语言的函数                         | 123 |
| 4.1 函数无所不在                            | 124 |
| 4.1.1 C 语言的函数                         | 124 |
| 4.1.2 为什么需要函数                         | 126 |

|       |                          |     |
|-------|--------------------------|-----|
| 4.1.3 | 种芝麻得西瓜                   | 129 |
| 4.1.4 | 细说函数声明                   | 135 |
| 4.1.5 | 数组作为函数参数                 | 136 |
| 4.1.6 | 对变量的又一次分类                | 142 |
| 4.1.7 | 递归——函数的自我调用              | 153 |
| 4.1.8 | 内部函数与外部函数                | 158 |
| 4.1.9 | main 函数                  | 159 |
| 4.2   | 预处理——粮草先行                | 160 |
| 4.2.1 | 拿来主义者 include            | 160 |
| 4.2.2 | 好好利用 define 提高程序可读性      | 162 |
| 4.2.3 | 灵活的条件编译                  | 165 |
| 4.3   | 小结                       | 168 |
| 4.4   | 习题                       | 169 |
| 第 5 章 | C 语言的指针                  | 171 |
| 5.1   | 指针不过是另外一种数据类型            | 172 |
| 5.1.1 | 从地址开始谈起                  | 172 |
| 5.1.2 | 指针变量和指针所指变量              | 173 |
| 5.1.3 | 正确书写指针自加减                | 181 |
| 5.2   | 指针与数组——唇齿相依              | 183 |
| 5.2.1 | 指向一维数组的指针                | 183 |
| 5.2.2 | 指针也有更高维数                 | 191 |
| 5.2.3 | 小心指针数组                   | 196 |
| 5.2.4 | **p                      | 199 |
| 5.2.5 | 动态数组的实现                  | 201 |
| 5.3   | 指针使函数充满灵性                | 202 |
| 5.3.1 | 函数可以有指针吗                 | 202 |
| 5.3.2 | 理解函数返回的指针                | 211 |
| 5.3.3 | 函数的参数传递                  | 212 |
| 5.4   | 小结                       | 215 |
| 5.5   | 习题                       | 216 |
| 第 6 章 | struct、union 以及 enum     | 219 |
| 6.1   | struct——组合数据类型 DIY       | 220 |
| 6.1.1 | struct 类型不等于 struct 类型变量 | 220 |
| 6.1.2 | 使用自定义的 struct 类型变量       | 224 |
| 6.1.3 | 一连串 struct 是什么样的         | 226 |
| 6.2   | 指针，又见指针                  | 228 |
| 6.2.1 | 通过指针使用 struct 变量         | 228 |

|                                 |            |
|---------------------------------|------------|
| 6.2.2 struct 作为函数的参数和返回值        | 231        |
| 6.3 链表——另一种串式结构                 | 232        |
| 6.4 union——共用体类型                | 240        |
| 6.5 enum——枚举类型                  | 244        |
| 6.6 使用 typedef 定义类型             | 247        |
| 6.7 小结                          | 248        |
| 6.8 习题                          | 249        |
| <b>第 7 章 C 语言的位运算</b>           | <b>251</b> |
| 7.1 无规则不成方圆——数据的编码规则            | 252        |
| 7.1.1 字节和位                      | 252        |
| 7.1.2 原码                        | 252        |
| 7.1.3 反码                        | 253        |
| 7.1.4 补码                        | 253        |
| 7.2 位运算的 6 种武器                  | 255        |
| 7.2.1 “按位与”运算符&                 | 255        |
| 7.2.2 “按位或”运算符                  | 256        |
| 7.2.3 “按位异或”运算符^                | 257        |
| 7.2.4 “取反”运算符~                  | 258        |
| 7.2.5 “左移”运算符<<                 | 258        |
| 7.2.6 “右移”运算符>>                 | 259        |
| 7.2.7 不同长度的数据进行位运算              | 259        |
| 7.3 哪里需要使用位运算                   | 260        |
| 7.4 小结                          | 262        |
| 7.5 习题                          | 262        |
| <b>第 8 章 C 语言的文件操作</b>          | <b>263</b> |
| 8.1 把文件看作一种数据类型                 | 264        |
| 8.2 芝麻开门芝麻关门                    | 265        |
| 8.2.1 打开文件的 fopen               | 265        |
| 8.2.2 关闭文件的 fclose              | 265        |
| 8.3 对文件进行读和写                    | 266        |
| 8.3.1 fgetc、fputc、fgets 和 fputs | 267        |
| 8.3.2 fscanf 和 fprintf          | 273        |
| 8.3.3 fread 和 fwrite——最常用的函数    | 277        |
| 8.3.4 fseek——步步为营               | 282        |
| 8.4 小结                          | 285        |
| 8.5 习题                          | 285        |

|                            |     |
|----------------------------|-----|
| 第9章 像程序员那样 Coding .....    | 287 |
| 9.1 首先让程序看上去很美 .....       | 288 |
| 9.1.1 { }与对齐 .....         | 288 |
| 9.1.2 空格和空行 .....          | 289 |
| 9.1.3 合理的长度 .....          | 291 |
| 9.1.4 规范的注释 .....          | 291 |
| 9.2 名不正自然言不顺 .....         | 292 |
| 9.2.1 匈牙利命名法 .....         | 293 |
| 9.2.2 变量的命名 .....          | 293 |
| 9.2.3 函数与文件的命名 .....       | 294 |
| 9.2.4 用宏替代 .....           | 295 |
| 9.3 一些可能会碰到的问题 .....       | 295 |
| 9.3.1 switch 和 break ..... | 295 |
| 9.3.2 比较问题 .....           | 296 |
| 9.4 指针的陷阱 .....            | 297 |
| 9.4.1 数组和指针 .....          | 298 |
| 9.4.2 指针和内存 .....          | 298 |
| 9.5 小结 .....               | 300 |
| 9.6 习题 .....               | 300 |
| 附录 C 语言常用函数参考 .....        | 301 |
| A.1 字符串函数 .....            | 301 |
| A.2 输入/输出函数 .....          | 302 |
| A.3 数学函数 .....             | 304 |
| A.4 屏幕和图形函数 .....          | 305 |
| A.5 时间/日期函数 .....          | 306 |
| A.6 转换函数 .....             | 306 |
| A.7 内存函数 .....             | 306 |
| A.8 接口/系统调用函数 .....        | 307 |

# 第 0 章

## C 语言基础

工欲善其事，必先利其器。

——《论语》

学完本章，你将会：

- ◆ 了解 C 语言的历史
- ◆ 了解常用的 C 语言的集成开发环境
- ◆ 会写一个最简单的 C 程序

## 0.1 第二天，上帝创造了 C

不要说没有接触过 C 语言，它可能正在为你默默服务着。普通计算机使用的 Windows 操作系统基本上是 C 语言写成的，用来进行文档处理的 Word、Excel 也是用 C 语言写成的，甚至在手机、微波炉的芯片中，都有 C 语言踪影。事实上，在计算机领域的每一个角落，一般都能够发现 C 语言的足迹。在这个时代，C 语言不是唯一的程序设计语言，但却一定是影响力最大和被认为最有存在价值的程序设计语言。有时甚至怀疑，是不是上帝假借了人之手，创造了 C 语言。

在这里简要地回顾一下 C 语言的产生和发展过程。

(1) 1967 年，英国剑桥的 Martin.Richards 在以前的 CPL (Combined Programming Language) 语言的基础上进行简化推出了 BCPL (Basic Combined Programming Language)。

(2) 1969 年，美国贝尔实验室的 K.Thompson 和 D.M.Ritchie 成功使用汇编语言——开发出 UNIX 操作系统的第一版本，在 DEC PDP-7 微机上实现。

(3) 1970 年，K.Thomapson 进一步简化了 BCPL 语言，设计出一种贴近硬件的语言——B 语言。并使用 B 语言再一次写了 UNIX 操作系统，仍旧在 PDP-7 上实现；一年以后在 PDP-11/20 上实现了 B 语言。

(4) 1973 年，D.M.Ritchie 在 B 语言基础上改进，增强功能并引进数据类型而设计出 C 语言。同年 D.M.Ritchie 和 K.Thomapson 一起使用 C 语言改写 UNIX 系统。改写后的 UNIX 系统有 90% 以上是用 C 语言写成的。

(5) 1977 年，《可移植 C 语言编译程序》的出现使得 C 语言移植到其他机器时所需做的工作得到很大简化。随着 UNIX 系统在不同硬件平台上的广泛应用，C 语言被移植到大量各种类型的计算机上，并最终脱离 UNIX 和 PDP 平台。

从上面的 C 语言产生过程可以看到，C 语言最初是为了更好地实现 UNIX 操作系统而发展来的，因而它很自然地具有实现操作系统所必须的汇编语言的一些特点。

(1) 生成的可执行代码简短而高效。用 C 语言编写的程序，其效率大约是实现相同功能的 C++ 代码的 8~10 倍，仅比汇编语言效率低 10%~20%，但是 C 语言的可理解性和易维护性却大大高于汇编语言，这一特点使它特别适合应用于系统资源有限的应用场合，如嵌入式系统。

(2) 可以直接进行低级的操作。C 语言可以用来实现汇编语言的大部分功能，如位 (bit) 操作、直接对硬件端口进行读写等。使它可以用来编写与硬件较接近的系统软件。

(3) 指针的实现。C 语言中使用指针的概念实现对内存单元的访问控制。指针本身是一个可以被赋值和进行运算的变量，它指向一个固定长度的内存单元（可以把指针想象成这个内存单元的门牌号码），对其所指向的内存单元及其邻近单元的访问，可以通过对指针简单地加減运算而得到。

与此同时，C 语言也兼具高级语言的特点，主要表现在以下几个方面。

- 具有丰富的数据类型和运算符。可以方便地用来实现各种复杂数据结构的运算。
- 可移植性好。几乎可以在大多数通用的计算机软硬件平台上不加修改地运行。

- 具有结构化的程序流控制语句，同时引入函数的概念来实现程序的模块化，是结构化程序设计的理想语言。

- 语法限制不太严格，程序设计自由度大。与一般的高级程序设计语言不同，C 语言的语法检查比较宽松。虽然近年来软件工程的不断规范化和标准化，为了保证较大的项目在编程风格上的统一和减少出错率，许多商业化的语法检查器也逐渐严格起来，但总的来说 C 语言的语法检查比起其他高级语言来还是比较宽松的，允许程序员灵活地运用一些非常规的技巧。

由于 C 语言同时兼具高级语言和低级语言的特点（因而 C 语言也被称为“中级语言”），这使它具有其他高级语言无法相比的速度优势和灵活性，而它简单的语法规则和易读性又弥补了比它更加底层的汇编语言繁琐、难以调试和难以理解的缺点，这是 C 语言被广泛推崇的原因之一，而这些优点仅是 C 语言具有种种优秀特点的一部分。

“真正的程序员用 C”，为什么不早点掌握它，来和计算机实现沟通呢？

## 0.2 应该准备什么

C 语言几乎可以在现在通用的任何一种计算机平台上进行编译和运行。一般情况下，只要有一台计算机，在这台计算机上运行着 DOS、Windows、UNIX 或者 Linux 的任何一个现存版本，那么就有了一个可以学习或使用 C 语言的基础环境了。

使用 C 语言离不开 C 语言的编程环境，或者称为开发系统。如果计算机使用的是 DOS，那么可以使用 Turbo C 或 Borland C 系列集成开发系统，它们很容易使用。如果使用的操作系统是 Windows98 或更新的 Windows 版本，仍然可以使用它们，而且会发觉它们在计算机中运行得很好。更奢侈的方式是使用 Microsoft 的 Visual C++ 集成开发环境。所以说奢侈是因为 Turbo C 和 Borland C 的很多版本都可以在网上免费获得，而一套 Visual C++ 的开发系统是需要比较大投资的，而且使用它来编制 C 程序绝对是杀鸡用牛刀。

在 UNIX 或 Linux 下，需要一个好用的文本编辑器，比如 vi，一个 gcc 或者其他任何 C 语言编译工具。但总得来说，在 UNIX 或 Linux 环境下，需要更多的专业知识。

## 0.3 热身运动

编写程序永远是一门必须在实践中不断摸索才能逐渐掌握的技术。当然，所有的技术，无论是木工做家具还是软件工程师编写一个庞大的系统，都是需要在实践中不断摸索的。相比其他的技术实践，编写程序对于实践者本身客观条件的要求可能是最低的。一个人是否有发达的臂力或过人的视觉并不能对其所编写的程序产生任何实质性的影响。只要能找到软件和硬件条件都符合 0.2 节中描述的计算机，或者任何一台能够提供 C 语言程序编译工具和运行环境的计算机就可以进行 C 语言的编程工作。本书每一章的后面都提供了一定的习题，在阅读这本书的过程中，最好能够不断地在计算机上进行编程和调试工作，这样才是正确而高效的学习 C 语言的过程。如果暂时无法上机，那么非常诚恳地建议读者将编写的程序源代码写在纸上或者用打字机打出来，上机时再将程序输入计算机进行调试。

下面以一个简单的“hello world”程序为例，说明在两种不同环境下进行 C 语言上机调试的操作过程。请读者按照自己的实际条件选择一种环境按照下述步骤进行一下热身。

在叙述前先对下面将涉及到的英文缩写做一个阐述。

IDE: Integrated Development Environment 的缩写，中文称它为集成开发环境。集成开发环境是一个整合了文本编辑工具（就是输入程序的地方）、语法检查器（它检查程序有没有错误）、编译器、连接器和函数库等工具的软件环境。在 IDE 中，开发人员不必再像早期的程序员那样一次次地重复手动输入命令和参数进行程序的开发和调试，而只需一次性设置好参数，在使用过程中点击相应的菜单项即可调用已经设置好的命令和参数对程序进行操作，大大简化了程序的调试过程。现在无论在 DOS 还是 Windows 操作系统下都有大量成熟的 IDE 可供选用。随着 UNIX 下 X windows 技术的成熟，现在在 UNIX 下也可以使用直观的图形界面 IDE 进行程序的开发了。



提示：不要担心才看了这么几页就有这么多的新名词，如果不了解它们，也不必担心，随着学习 C 语言的深入，会自然而然地了解并掌握它们。

### 1. TC++ 3.0 IDE 或者 TC 2.0 IDE

这是要介绍的第一个 IDE，它的全称 Turbo C/C++ Integrated Development Environment，它的 2.0 版本的 IDE 几乎可以在任何 C 语言编程的相关网站方便地下载到。TC++ 3.0 IDE 是 Turbo C 2.0 IDE 的升级版本，它不仅一如既往地、很好地支持 ANSI C 编译标准，而且添加了对 C++ 的支持。这意味着不仅能在 TC++ 3.0 IDE 中编译 C 程序而且可以方便地编译 C++ 程序。但是在本书中只使用它和 C 语言有关的功能。如果只是想掌握一门语言，那么 TC++ 3.0 是最理想的，它小且简单，提供了所需的全部功能。

下面说明 TC++ 3.0 的安装过程。

(1) 将下载的 Borland Turbo C++ 3.0.zip 文件解压到硬盘上的一个目录，比如 E:\TC30。打开该目录，找到文件 INSTALL.exe，如图 0-1 所示。

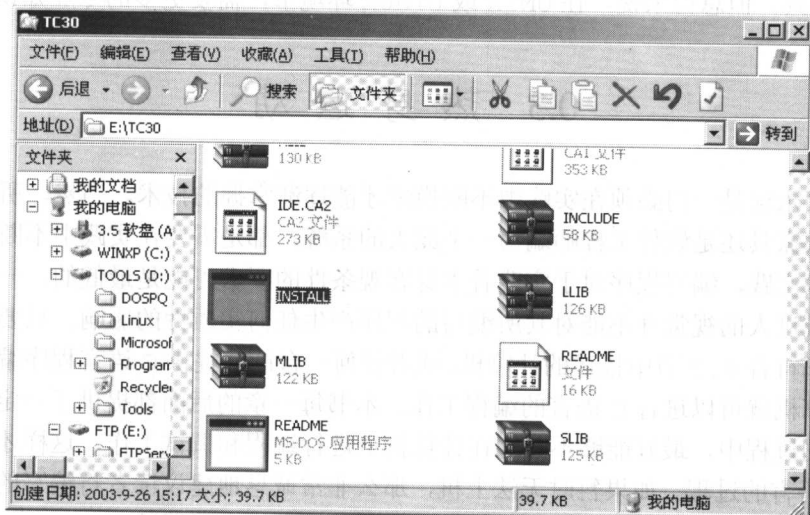


图 0-1 找到 INSTALL.exe 文件

(2) 双击 INSTALL.exe 文件, 打开如图 0-2 所示的安装界面, 提示安装程序将需要 10.5MB 的硬盘空间来安装所有的组件, 按回车键【Enter】即可进行安装, 按【ESC】键则会退出安装程序。

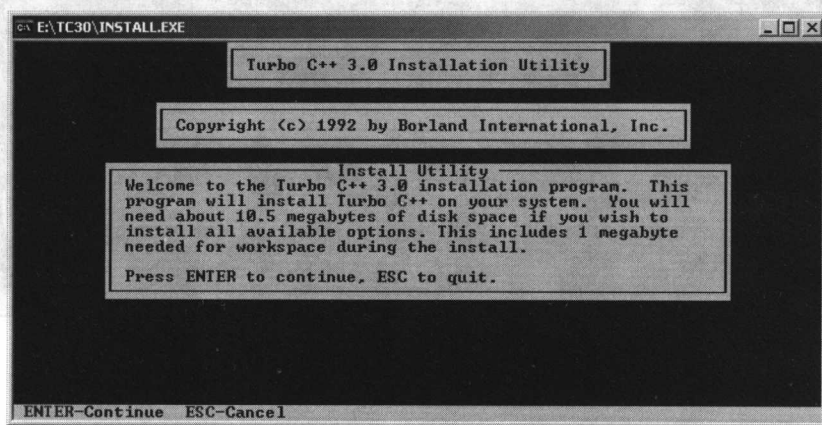


图 0-2 TC++3.0 安装界面

(3) 按【Enter】键后, 出现如图 0-3 所示的界面, 要求输入安装程序所在的驱动器名, 这里输入 E 后按【Enter】键 (因为安装文件在 E 盘), 出现如图 0-4 所示的设置安装程序所在路径的界面。

(4) 输入 \TC30 (安装文件所在的目录) 后按【Enter】键, 出现如图 0-5 所示的安装选项设置界面。在该界面中, 可以选择安装路径和安装的可选组件。按方向键上【↑】、下【↓】键可以选择【Directories...】或【Options...】。

(5) 选择【Direcotries...】并按【Enter】键后, 出现如图 0-6 所示的界面, 按上【↑】下【↓】键可选择设置不同组件的安装目录, 比如选择【Tubor C++ Directory】后, 按【Enter】键, 出现如图 0-7 所示的设置安装目录界面, 输入所要安装的目录, 按【Enter】键即可。

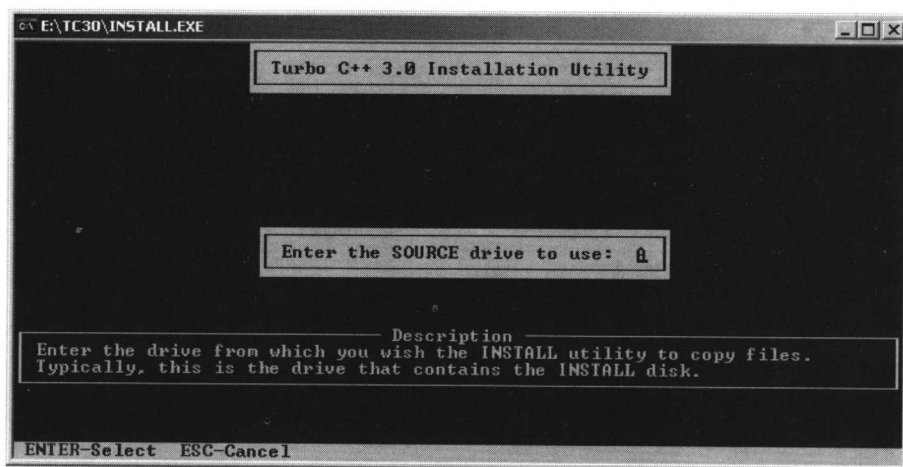


图 0-3 选择安装程序所在驱动器

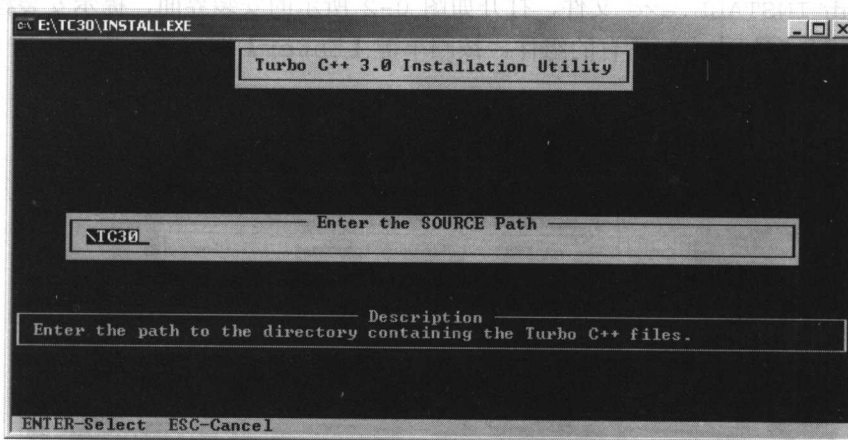


图 0-4 设置安装程序路径

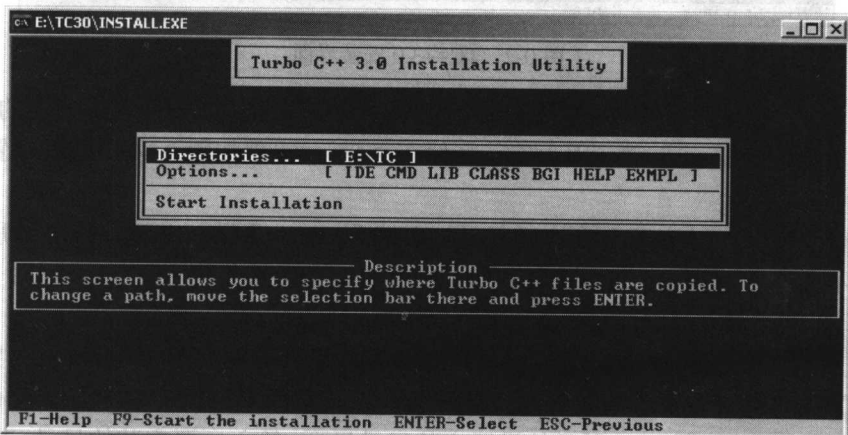


图 0-5 安装选项设置

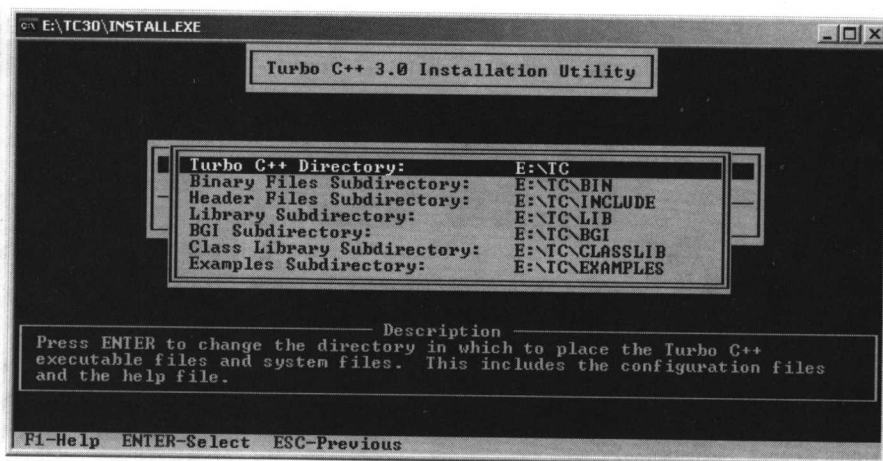


图 0-6 安装目录选项

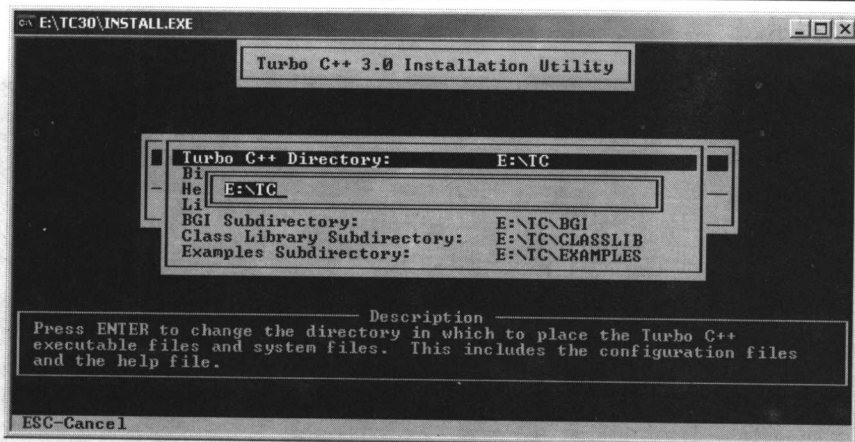


图 0-7 设置安装目录

这里的安装目录都选为 E:\TC 目录以及相应的 Include 目录、Lib 目录等。返回如图 0-5 所示的界面后，将选择条移动到【Options...】选项，按【Enter】键，出现如图 0-8 所示的选择安装组件的界面。在该界面中，可选择不同的安装组件。

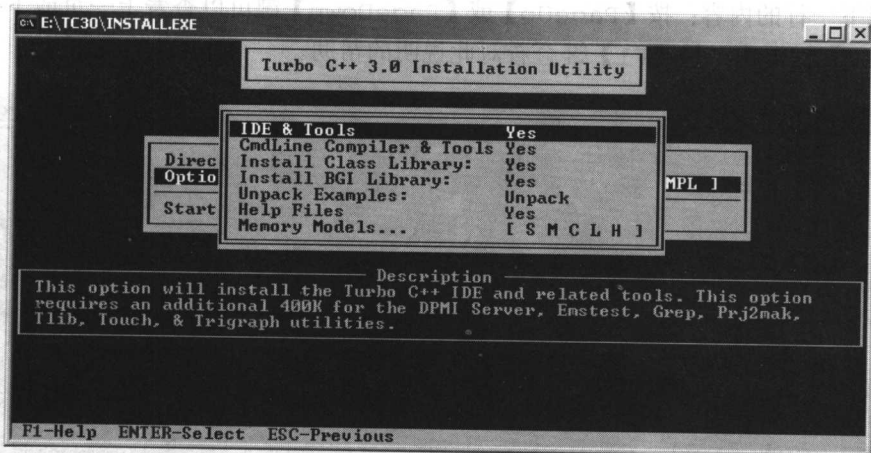


图 0-8 安装选项

- IDE & Tools——安装 Turbo C++ IDE 和相关的工具。
- CmdLine Compiler & Tools——安装 Turbo C++ 命令行编译器和相关工具。
- Install Class Library——安装 Turbo C++ 类库。
- Install BGI Library——安装 BGI (Borland Graphic Interface, 用于操作图形的) 库。
- Unpack Examples——安装 C 和 C++ 例子程序和源代码，Unpack 选项说明安装解压后的文件，Packed 则安装未解压的 zip 文件，需要用户以后自己再解压缩。
- Help Files——安装 IDE 和命令行的帮助文件。
- Memory Models...——安装内存模式的相关组件。

(6) 这里都选择默认设置，返回如图 0-5 所示的界面，移动选择条到【Start Installation】，按【Enter】键，安装程序就开始安装，一般很快就能安装完，出现如

图 0-9 所示的提示。

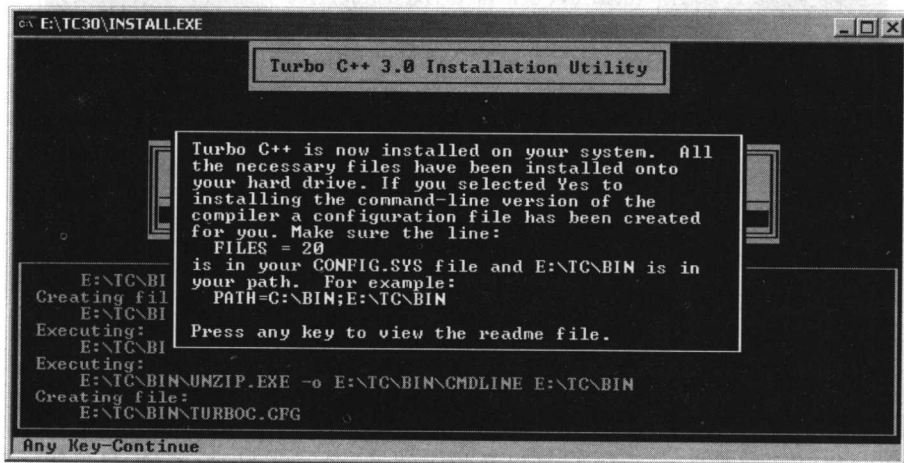


图 0-9 安装结束的提示

(7) 按任意键后, 出现如图 0-10 所示的说明。查看说明时, 可以按【↑】或【↓】键查看上一行或下一行的内容; 按【PageUp】或【PageDown】键可以查看上一页或下一页的内容; 按【ESC】键退出; 按【F1】键查看对此说明的帮助。

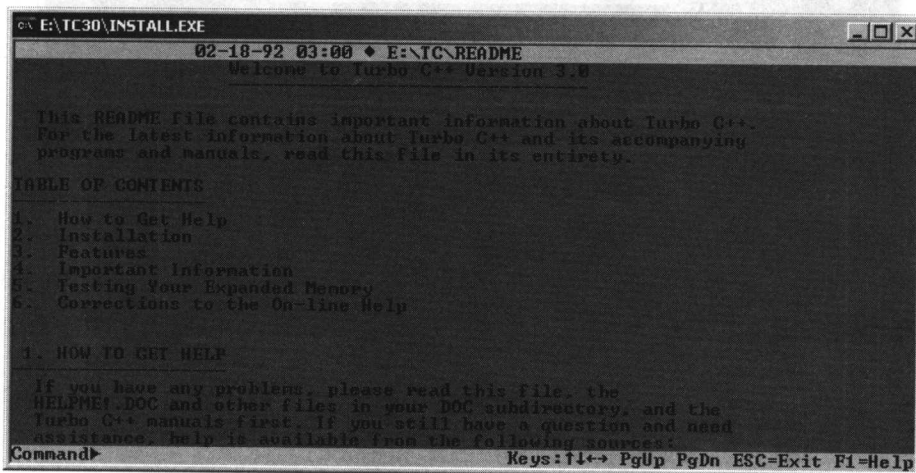


图 0-10 Turbo C++ 3.0 的说明文件

至此, Turbo C++ 3.0 的安装就已经完成, 下面就可以运行程序, 编写第一个 C 程序了。

下面的步骤将启动 Turbo C++ 3.0, 并编写 Hello world 程序。

(1) 打开 TurboC++3.0 的安装目录(这里是 E:\TC)下的【Bin】目录, 找到 TC.exe 文件, 双击该文件启动 Turbo C++ 3.0 IDE, 将在屏幕上出现如图 0-11 所示界面。

如图 0-11 所示, 选择顶端菜单【File】|【New】选项建立新文件。而一般在 TurboC++ 3.0 IDE 启动时系统会自动建立一个默认名称为“NONAME00.cpp”的新文件, 可以直接在这个文件中输入如下程序段: