



Beginning Programming

程序设计入门经典

(美) Adrian Kingsley-Hughes 著
Kathie Kingsley-Hughes 译
顾晓锋



清华大学出版社

TP311.1

39

程序设计入门经典

(美) Adrian Kingsley-Hughes 著
Kathie Kingsley-Hughes 译
顾晓锋 译

清华大学出版社

北 京

Adrian Kingsley-Hughes, Kathie Kingsley-Hughes

Beginning Programming

EISBN: 0-7645-8406-5

Copyright © 2005 by John Wiley & Sons, Inc.

All Rights Reserved. Authorized translation from the English language edition published by John Wiley & Sons, Inc.

本书中文简体字版由 John Wiley & Sons, Inc. 授权清华大学出版社出版。未经出版者书面许可, 不得以任何方式复制或抄袭本书内容。

北京市版权局著作权合同登记号 图字: 01-2005-3307

版权所有, 翻印必究。举报电话: 010-62782989 13501256678 13801310933

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。

本书防伪标签采用特殊防伪技术, 用户可通过在图案表面涂抹清水, 图案消失, 水干后图案复现; 或将表面膜揭下, 放在白纸上用彩笔涂抹, 图案在白纸上再现的方法识别真伪。

图书在版编目(CIP)数据

程序设计入门经典/(美)金斯利-休斯(Kingsley-Hughes, A.), (美) 金斯利-休斯(Kingsley-Hughes, K.)著; 顾晓锋 译.

—北京: 清华大学出版社, 2006.7

ISBN 7-302-12548-1

I. 程… II. ①金… ②金… ③顾… III. 程序设计 IV. TP311.1

中国版本图书馆 CIP 数据核字(2006)第 009823 号

出版者: 清华大学出版社

地址: 北京清华大学学研大厦

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

邮编: 100084

社总机: 010-62770175

客户服务: 010-62776969

组稿编辑: 曹 康

文稿编辑: 李 阳

封面设计: 孔祥丰

版式设计: 康 博

印刷者: 清华大学印刷厂

装订者: 北京市密云县京文制本装订厂

发行者: 新华书店总店北京发行所

开 本: 185 × 260 印张: 23 字数: 589 千字

版 次: 2006 年 7 月第 1 版 2006 年 7 月第 1 次印刷

书 号: ISBN 7-302-12548-1/TP · 8027

印 数: 1 ~ 4000

定 价: 39.90 元

前言

随着计算机在单位、家庭、图书馆以及学校中的普及，先前的那些计算机初级用户必然想成为更高层次的计算机用户，他们希望学习如何用新的方式让计算机为他们工作。很快他们认识到要实现这个愿望，应先学习如何成为一名编程人员。而要想成为一名编程人员，首先是要挑选一本类似于本书的参考书。

目前大多数关于编程的图书都存在一个问题，即认为读者：

- 阅读该书之前已经阅读了大量有关编程的资料。
- 阅读该书之前已经具有一些编程的背景知识。

读者对象

大多数计算机编程图书主要面向那些已具有一定编程基础并且希望深入研究编程方法的读者。这些读者了解函数和数组之间的区别，明白优秀的编程人员都会声明变量，也会使用注释。

本书与那些书不同，主要面向那些没有任何编程背景知识的读者，他们希望通过学习成为编程人员。这些读者包括教师、学生、护士、律师、卡车司机和飞行员等。他们希望自己操作计算机的能力能够成为他们所拥有的一项优势。

以前，编程图书主要关注那些希望成为职业编程人员的读者，而本书适合每一位想学习编程的读者。

同时我们也非常高兴立志成为专业编程人员的读者能阅读本书。本书提供了扎实的理论基础，使读者可以缩小自己的学习范围，专攻特定的编程语言或编程任务。书中并没有出现“如何在 15 分钟内成为一名专业编程人员”或“如何编写一个举世无双的应用程序”这些字眼，但通过阅读本书读者均会发现，本书提供了扎实的编程知识，使读者能在编程领域中掌握相应的技术。

内容范围

本书的目的是使那些对编程感兴趣的读者掌握所需的技术和实践经验，从而实现他们的目标。因此，本书重点介绍编程人员所需的各种技术，以及编程的理论，并把理论和编写实际代码相结合，另外使用多种编程语言来演示和解释这些内容。

本书使用的语言包括：

- C++
- Java
- VBScript

- JavaScript

本书提供了用这些语言编写的大量代码示例以及各种编程工具和编译器。

结构安排

本书循序渐进地介绍了精通编程技术所需的学习过程，这些技术是优秀的编程人员应该掌握的。推荐从第1章开始，逐章学习。本书的结构安排如下：

- **第1章 编程概述：**介绍了编程的概念以及这些概念对编程人员的意义。
- **第2章 学习编程的原因：**细述了为什么要学习编程的种种理由以及每个人成为编程人员所采用的不同方法。
- **第3章 计算机读取代码的方式：**讨论了计算机如何保存和处理代码，力图让读者了解计算机的后台工作。
- **第4章 从概念到代码——代码语言：**讨论计算机语言基础以及文本和数字在数字化领域中如何表示。这一章详细介绍了二进制和十六进制数字系统如何构成代码的主体部分，以及ASCII中的数字式字符如何用二进制表示。
- **第5章 编程工具：**介绍运行源代码和成为优秀的编程人员所需的工具。在本章会发现，文本编辑器、编译器以及其他工具可以使编码的整个处理过程更简单、更轻松。
- **第6章 简单编码：**通过介绍编写真正能执行任务的代码，向读者介绍各种编程语言中相互一致的许多关键概念。
- **第7章 编码结构：**深入研究如何在代码中仅添加几行代码，使代码形成结构，让代码能根据各种不同的输入执行任务，允许根据测试条件来运行某些语句代码。
- **第8章 解决问题：**编码就是为了解决问题。本章介绍代码如何出色地完成问题求解的过程，以及如何把编码过程分解成一系列更短更易于掌握的步骤。
- **第9章 调试：**每次编写的代码都有可能会出现错误，本章主要介绍编码中会出现的各种错误以及如何发现并改正这些错误。
- **第10章 界面：**编码的同时还需注意如何将所编写的程序展示给终端用户。本章提供了如何使应用程序易用的一些有用的技巧和实践经验。
- **第11章 详述编程项目：**目前为止，已介绍了编程的各个阶段。本章综合运用之前所讲的内容，让读者掌握从概念开始到完成产品的整个编程项目，让读者有机会在从头至尾的编程过程中，综合运用各种编程技能。
- **第12章 操作文件：**介绍文件系统的操作实践，如何创建、修改和删除文件和文件夹。
- **第13章 注册表：**Windows注册表是设置应用程序的主要存储区域，本章介绍注册表的导航，以及如何编写访问、读取和修改注册表的代码。
- **第14章 组织、计划和版本控制：**如果不想在将来的编程过程中出现棘手问题，跟踪项目和代码的进度是很重要的。本章主要讨论如何控制源代码和如何处理将创建的文件。
- **第15章 编译代码和编译方法：**本章讲述代码编译过程以及编译代码给编程人员所带来的优势。
- **第16章 发布项目：**这是本书的最后一章，介绍向终端用户发布代码或应用程序的方法，可以利用书中提出的多种方案将代码提供给所需的用户。

所需工具

本书涉及到的所有工具和资料都很容易在 Internet 网上获得或已经安装在计算机上。

编写代码时主要使用文本编辑器，Windows 系统中的记事本就是一个理想的文本编辑器。但如果希望得到更多的功能，可以选用其他工具，这些工具有些是免费的，而有些需要通过商业途径得到。

本书使用的是 UltraEdit 工具，它功能强大，可以到 www.ultraedit.com 网站下载商业软件包。对编程人员来讲，这是最好的一种文本编辑工具。虽然要收费，但功能很强大。如果不想使用这个工具，也可以用其他免费工具。

另外还需用到编译器。本书使用的主要编译器是免费的 Borland C++ 编译器，读者可以到 www.borland.com/products/downloads/download_cbuilder.html 网站下载该编译器。

其他工具将会在用到时再介绍。

源代码

对于本书中的示例代码，读者可以选择手工输入所有代码，也可以使用本书所附的源代码文件。本书中的所有源代码都可以从 www.wrox.com 或 www.tupwk.com.cn/downpage 网站下载。登录 www.wrox.com 网站后，找到本书书名(可以使用搜索框，或使用书名列表)，单击 Download Code 链接，就可以下载本书的所有源代码。

因为有许多相同的书名，所以可以用最简单的搜索方式，即用本书英文版的 ISBN 进行搜索，ISBN 号为：0-7645-8406-5。

下载完代码后，可以用您最喜欢的解压缩工具对它解压缩。另外，也可以访问 Wrox 的下载主页：www.wrox.com/dynamic/books/download.aspx 获得本书以及 Wrox 公司其他图书的源代码。

勘误表

尽管我们已经尽了各种努力来保证本书不出现错误，但是错误总是在所难免，如果您在本书中找到了错误，例如拼写错误或代码错误，请告诉我们，我们将不胜感激。通过勘误表，可以让其他读者避免受挫，当然，这还有助于提供更高质量的信息。

要在网站上找到本书的勘误表，可以登录 <http://www.wrox.com>，通过 Search 工具或书名列表查找本书，然后在本书的信息页面上，单击 Book Errata 链接。在这个页面上，可以查看已经提交并通过编辑检查的所有勘误。包含每本书的错误勘误表链接的完整图书列表可通过 <http://www.wrox.com/misc-pages/booklist.shtml> 获得。

如果在 Book Errata 页面上没有您发现的错误，请您将发现的错误发送到 fwkbook@tup.tsinghua.edu.cn。我们会检查您的信息，如果正确，就把它上传到该书的勘误表页面上，或在本书的后续版本中采用。

p2p.wrox.com

P2P 邮件列表是为作者和读者之间的讨论而建立的。读者可以在 p2p.wrox.com 上加入 P2P 论坛。该论坛是一个基于 Web 的系统，用于传送与 Wrox 图书相关的信息和相关技术，与其他读者和技术用户交流。该论坛提供了订阅功能，当论坛上有新贴子时，会给您发送您选择的主题。Wrox 作者、编辑和其他业界专家和读者都会在这个论坛上进行讨论。

在 <http://p2p.wrox.com> 上有许多不同的论坛，帮助读者阅读本书，在读者开发自己的应用程序时，也可以从这个论坛中获益。要加入这个论坛，需执行下面的步骤：

- (1) 进入 p2p.wrox.com，单击 Register 链接。
- (2) 阅读其内容，单击 Agree 按钮。
- (3) 提供加入论坛所需的信息及愿意提供的可选信息，单击 Submit 按钮。

然后就可以收到一封电子邮件，其中的信息描述了如何验证账户，完成加入过程。

提示：

不加入 P2P 也可以阅读论坛上的信息，但只有加入论坛后，才能发送自己的信息。

加入论坛后，就可以发送新信息，回应其他用户的贴子。可以随时在 Web 上阅读信息。如果希望某个论坛给自己发送新信息，可以在论坛列表中单击该论坛对应的 Subscribe to this Forum 图标。

对于如何使用 Wrox P2P 的更多信息，可阅读 P2P FAQ，了解论坛软件的工作原理，以及许多针对 P2P 和 Wrox 图书的常见问题解答。要阅读 FAQ，可以单击任意 P2P 页面上的 FAQ 链接。

目 录

第 1 章 编程概述..... 1	第 3 章 计算机读取代码的方式..... 22
1.1 编程的发展过程..... 1	3.1 读取代码..... 22
1.2 编程的概念..... 3	3.1.1 “自顶向下”的处理过程..... 22
1.2.1 存在多种编程语言的原因..... 4	3.1.2 分解代码..... 24
1.2.2 不同的代码, 相同的结果..... 5	3.1.3 语句..... 28
1.3 能编程的程序..... 9	3.1.4 函数/过程..... 29
1.3.1 开发环境..... 9	3.2 编程中的语句和段落..... 30
1.3.2 编译器..... 9	3.2.1 代码行..... 30
1.4 小结..... 9	3.2.2 代码段落..... 31
第 2 章 学习编程的原因..... 11	3.3 数据存储..... 31
2.1 编程的意义..... 11	3.4 小结..... 34
2.1.1 “职业”编程人员..... 11	第 4 章 从概念到代码——代码语言..... 35
2.1.2 解决问题..... 14	4.1 二进制..... 35
2.1.3 因为想作贡献..... 15	4.1.1 二进制介绍..... 35
2.1.4 仅为娱乐..... 15	4.1.2 大数..... 37
2.1.5 名望..... 15	4.1.3 按位分组..... 38
2.2 学习的内容..... 15	4.1.4 二进制数学..... 41
2.3 对编程的看法和实际情况..... 19	4.1.5 需要二进制的原因..... 44
2.3.1 看法 1——编程工具需 花费许多钱..... 19	4.2 十六进制..... 44
2.3.2 看法 2——成为编程人员 须具备计算机科学学位..... 19	4.2.1 十六进制介绍..... 44
2.3.3 看法 3——学习编程需 花数年时间..... 19	4.2.2 十六进制和 Windows Calculator..... 46
2.3.4 看法 4——编程只适合年青人..... 20	4.3 字符表示..... 46
2.3.5 看法 5——编程需要顶级 的超级计算机..... 20	4.4 运算符..... 54
2.3.6 看法 6——编程会上瘾!..... 20	4.4.1 算术运算符..... 54
2.3.7 看法 7——编程语言一直 在改变..... 20	4.4.2 赋值运算符..... 55
2.3.8 看法 8——一旦学会了某种 编程语言, 学习其他语言就 会简单些..... 20	4.4.3 比较运算符..... 55
2.4 小结..... 21	4.4.4 逻辑运算符..... 56
	4.4.5 字符串运算符..... 56
	4.5 小结..... 57
	第 5 章 编程工具..... 58
	5.1 建立自己的工作区..... 58
	5.1.1 键盘..... 58
	5.1.2 工作区..... 59
	5.1.3 办公桌..... 59

5.1.4 显示器	59	7.6.1 二维数组	148
5.2 选择编程语言	62	7.6.2 多维数组	149
5.2.1 学习编程	62	7.7 小结	150
5.2.2 学校/学院	62	第8章 解决问题	151
5.2.3 业务培训	63	8.1 解决问题的基础知识	151
5.2.4 业余编程人员	63	8.1.1 清楚需求	152
5.2.5 语言	64	8.1.2 研究	155
5.2.6 教您如何编程	64	8.1.3 把问题分解成小问题	157
5.3 工具	65	8.2 编写代码	158
5.3.1 通用工具和实用程序	65	8.3 小结	172
5.3.2 编程工具	70	第9章 调试	173
5.4 小结	76	9.1 人为错误	173
第6章 简单编码	77	9.2 错误, 错误, 还是错误	173
6.1 注释代码	77	9.3 编译器错误	174
6.1.1 VBScript 注释	77	9.4 运行库错误	186
6.1.2 Java Script 注释	80	9.5 逻辑错误	190
6.1.3 C++注释	82	9.6 发现错误	193
6.2 变量	83	9.6.1 输入代码后阅读每行代码	193
6.3 字符串	100	9.6.2 检查之前的语句	194
6.3.1 字符串概述	100	9.6.3 保持结构清楚	194
6.3.2 字符串操作	101	9.6.4 注释, 注释, 再注释	194
6.4 处理输入	105	9.6.5 删除代码的多义性	194
6.5 变量操作——执行简单的 数学功能	107	9.6.6 分号	195
6.6 小结	109	9.6.7 测试代码	196
第7章 编码结构	110	9.6.8 检查变量	196
7.1 结构的用途	110	9.7 小结	201
7.2 使用结构前的准备工作	111	第10章 界面	202
7.3 函数	120	10.1 界面概述	202
7.4 条件控制	126	10.2 界面的重要性	204
7.4.1 条件判断	126	10.2.1 什么是界面	204
7.4.2 条件	127	10.2.2 所有的软件都有界面吗	204
7.4.3 多条件控制	132	10.3 界面详述	205
7.5 循环	141	10.3.1 基于文本的界面	205
7.5.1 for 循环语句	141	10.3.2 程序纲要	207
7.5.2 死循环	143	10.3.3 正确提示输入	209
7.5.3 while 循环语句	143	10.3.4 对输出结果进行注释	212
7.5.4 do while 循环语句	144	10.3.5 对程序退出进行确认	213
7.6 数组	147	10.3.6 添加简单的帮助	214
		10.3.7 确认	219

10.4	非文本形式的界面	220	12.6.2	追加文件	260
10.5	按钮	220	12.6.3	以只读方式打开文件	261
10.5.1	菜单	222	12.6.4	ReadAll、ReadLine 和 Read 方法	262
10.5.2	复选框	223	12.7	删除文件和文件夹	265
10.5.3	单选按钮	224	12.7.1	删除文件	265
10.5.4	单行文本框	225	12.7.2	删除文件夹	265
10.5.5	多行文本框	225	12.8	小结	265
10.5.6	下拉式菜单	226	第 13 章	注册表	267
10.6	综合	227	13.1	Windows 注册表	267
10.6.1	简单应用程序	227	13.2	Windows 注册表概述	267
10.6.2	更多复杂的应用程序	230	13.2.1	定义	267
10.7	小结	232	13.2.2	Regedit 和 Regedit32	268
第 11 章	综述编程项目	233	13.2.3	备份注册表	269
11.1	计划编程项目	233	13.2.4	操作注册表	278
11.2	计划	234	13.3	通过编程来操作 Windowd 注册表	284
11.2.1	想法	234	13.3.1	用 VBScript 编辑注册表	284
11.2.2	要求	237	13.3.2	用 JScript 编辑注册表	288
11.3	编程阶段	240	13.4	Windows 注册表的使用	289
11.4	测试	242	13.5	使用注册表时应注意的问题	290
11.5	附加功能	245	13.6	小结	290
11.6	整理代码	245	第 14 章	组织、计划和版本控制	292
11.7	最终测试	246	14.1	组织	292
11.8	小结	246	14.1.1	组织自己	292
第 12 章	操作文件	247	14.1.2	组织自己的工作空间	292
12.1	保存数据的原则	247	14.2	主要事件——组织 自己的 PC 机	294
12.2	文件的生命周期	247	14.3	文件夹	295
12.3	文件	249	14.3.1	按语言分组	295
12.4	开始编程	249	14.3.2	按项目分组	296
12.5	用 VBScript 创建文件	249	14.3.3	文件夹内容注释	297
12.5.1	基本要素	250	14.4	文件名控制	299
12.5.2	创建文件夹	252	14.5	更多版本控制提示	300
12.5.3	创建多个文件	253	14.6	版本控制——查找其他版本	305
12.5.4	使用条件控制	253	14.7	软件版本控制	306
12.5.5	使用变量	255	14.8	小结	306
12.5.6	对文件名和文件夹名 进行提示, 以增加灵活性	255	第 15 章	编译代码和编译方法	307
12.5.7	检查相同文件	258	15.1	编译代码	307
12.6	编辑已存在的文件	258			
12.6.1	运行	260			

15.2	所有的编译器是否都相同.....	308	16.2	小结	343
15.2.1	错误处理	314	附录 A	术语表	344
15.2.2	不同的语言会怎样	318	附录 B	Web 资源	348
15.3	编译的作用	321	B.1	编程工具	348
15.3.1	保护智能属性	321	B.2	Java 工具	349
15.3.2	速度	321	B.3	Java 站点	350
15.3.3	功能提升	322	B.4	C++工具	351
15.3.4	安全性	322	B.5	C++站点	352
15.3.5	调试	322	B.6	BASIC 工具	352
15.4	编译代码的其他方法	323	B.7	BASIC 站点	353
15.5	小结	328	B.8	Web 脚本语言	354
第 16 章	发布项目	329	B.9	CD 刻录	354
16.1	发布类型	329	B.10	压缩工具	354
16.1.1	实体发布	329	B.11	其他站点	356
16.1.2	刻录光盘	333			
16.1.3	虚拟发布	340			

第 1 章 编程概述

如果您读过编程方面的图书，应当对编程的概念有相当清晰的了解。如果您不知道什么是编程，那么为什么想要学习编程呢？然而，许多刚接触编程的学生，甚至那些稍微懂得一点编程来从事具体工作，并从中积累经验的技术人员，也可能会从编程的发展过程、编程概念以及编程现状的概述中获益匪浅。

1.1 编程的发展过程

编程的发展过程比大多数人所想像的还要长久。许多人认为编程是从 20 世纪末才开始的，其实现代编程和编程语言可以追溯到 20 世纪 40 年代中期，比人们认为的早了近 60 年。

然而，在讲述 20 世纪 40 年代的编程历史之前，仍需再向前追溯到 1822 年，那时有个人叫 Charles Babbage。当他在英国剑桥大学读书时，偶然发现许多有关时间的计算设备(如天文图、潮汐图、航海图)在测量时，都存在临界误差，并且测量繁琐。这些误差造成许多船只、人员和货物在海上失踪。Charles Babbage 认为这些不精确是人的误差造成的，因此他想用蒸汽发动机来建立和维护那些图表。这个机器以及相配套的差分机(Difference Engine)消耗了 Babbage 余生的大部分时间。他甚至还向英国政府申请财政补助——多次要求政府给计算机科学研究提供资助。

研究了差分机 10 年以后，Babbage 意识到差分机是一个最终只能执行单操作的单用途机器，这是差分机的主要缺陷。Babbage 曾一度放弃了差分机，采用了更通用的解析机(Analytical Engine)。该解析机包括了现代计算机的基本组成部分，因此人们将 Babbage 称为“计算机之父”。由于 Babbage 遭遇病痛使得解析机没有得到广泛应用，几个世纪以来，编程人员和计算机科学家很遗憾未能形成一个清晰的文档来传达他的思想。

解析机的研究一直持续到 1842 年，苦于没有进展和结果，英国政府放弃了这一研究并取消了资助。但 Babbage 仍然继续研究解析机，直到 1847 年他重新研究差分机为止。1847-1849 年间，Babbage 完成了 21 张详细制图，这是差分机第二版本的引擎结构图。然而，Babbage 没有真正完成差分机和解析机。1871 年 Babbage 去世后，为了保存这些研究，他的儿子 Henry Prevost 复制了一些差分机的简单算术单元的副本，送到世界各地的各个研究机构，包括哈佛大学。

19 世纪的研究不断取得进展，1854 年，Charles Boole 描写了符号逻辑系统，并以他的名字命名(布尔逻辑)，一直沿用到现在(后面章节讲述)。该系统提出了逻辑术语，如大于、小于、等于、不等于，并建立了一套符号系统来描述这些术语。

俗话说“需要是发明之母”。1890 年美国国会给人口普查提出了更多的问题，使得这种需求达到了顶点。美国人口的不断增加意味着处理这些数据需要花费的时间越来越长。除非提高和加快处理数据的速度，否则估计 1890 年的人口普查数据到下一次普查即 1900 年还不能处理完！

因此，美国政府举办竞赛来引起人们对计算、传递数据处理设备研究的兴趣。Herman Hollerith 在比赛中获胜，在成功证明该项技术后，他继续把这项技术运用到其他国家的人口普查信息处理中。后来他创办了 Hollerith Tabulating 公司(这家公司与其他两家公司于 1914 年合作创办了 CTR 公司，即 Calculating Tabulating Recording Company。对于该公司您可能不太熟悉，但十年后该公司被命名为国际商用机器(International Business Machines, 简称 IBM)公司——毫无疑问您听说过该公司！)。

此后，研究进展似乎慢了些，到 20 世纪 20 年代中期，数字计算工具很少用于商业、工业或工程领域。事实上，最常用的是类似于计算尺的工具。然而情况在 1925 年开始转变，在美国麻省理工学院(MIT)，Vannervar Bush 建立了一个规模巨大的差分分析器，这种机器综合了积分和差分功能。大量的发明资金由洛克菲勒(Rockefeller)基金会提供，在 1930 年这台机器成为全世界最庞大的“计算机”。

编程发展过程中另一位主要人物是德国科学家 Konrad Zuse。1935 年，Zuse 研制了 Z-1 计算机，这台机器是第一台利用继电器和以二进制计算系统为基础的计算机，它是当代计算机时代的先驱。

Zuse 继续研究，于 1938 年在 Helmut Schreyer 的帮助下研制出 Z-2 计算机。他向德国政府申请财政资助，以用于计算机的发展和构建，由于要完成该项目的超出了预期战争结束的时间，因此他的申请遭到了拒绝。战争即将结束时，Zuse 逃避到 Hinterstein，之后来到瑞士，在瑞士苏黎世大学重建了 Z-4 机器。

Zuse 还发明了现代编程。1946 年他开发了世界上第一种编程语言——Plankalkül。他甚至还给 Z-3 计算机编写代码来与自己下国际象棋。这种语言具有开创性，很多现代语言中都包含了它，比如表格和数据结构。

Zuse 随后成立了计算机公司，这家公司后来被西门子公司合并。

1945 年，计算领域中另一重要的发展是单词 bug 的引入，这个单词几乎人尽皆知。1945 年，Grace Murray Hopper(后来成为 Hopper 上将)在哈佛大学研究 Mark II Aiken Relay Calculator 机器。该机器在那段时间一直出现问题，1945 年 9 月 9 号，一名技术人员发现一只小虫陷在电路之间(记录显示正好在继电器#70，面板 F 这个点上)。技术人员随即把这只小虫清除出来并把它贴在记录计算机使用和问题的日志上。日志上写着“发现第一个真正的小虫”。“调试机器”、“调试计算机”，以及后来出现的“调试计算机程序”等这些短语都是从这派生而来的。

尽管 Grace Hopper 一直很小心地澄清发生那件事的时候她不在场，但这是她经常乐于叙述的事件之一。

此后，研究进展加速发展。1949 年，发明了短代码(Short Code)。这种代码必须手工转换成机器可读代码(这个过程就是编译，compiling)，因此代码非常少，称之为短代码。

1954 年，IBM 开始开发 FORTRAN(FORMula TRANslator，公式转换器)语言。FORTRAN 语言可以方便地使用输入和输出系统，代码简短，便于掌握，因此在 1959 年发布后便立即取得成功(现在有些地方仍在使用它)。另外 FORTRAN 语言是第一个商业化的高级编程语言，由于它遵守人们熟悉的语法、方法和结构规则，因此代码很容易理解。

1958 年，FORTRAN II 和 ALGOL 发布后，接着又发布了 LISP 语言。直到 1959 年，另一种流行且使用长久的 COBOL(Common Business Oriented Language，面向商业的通用语言)语言诞生，它是在数据系统和语言会议上提出来的。COBOL 是一种主要使用在大型机器上的商业机构的语言，现在许多公司仍然在使用它。

其后,研究继续飞速发展,许多新语言和已有语言的其他版本不断发布。1968年间,Pascal语言开始出现,它于1970年发布。现在仍广泛用于教学。1970年还发布了另外两种语言,即Smalltalk和B-language,它们在计算方面具有革命性。Smalltalk语言是一种重要的语言,它是完全基于对象(书中会介绍)的语言,B-language语言的重要性则是它具有导向性。

B-language语言导致了什么语言呢?1972年,Dennis Ritchie在B-language语言的基础上开发出最终被称为C的语言(之前有段时间被称为NB)。C语言更加简单、有效、灵活,开创了编程的新时代,利用它可以尽可能用较少的代码实现较多的功能,比之前的B语言更快更简单。

1975年,Wong博士发布了TinyBASIC语言。TinyBASIC语言只占2KB的内存,通过纸带上载到计算机。这同样具有开创性,因为它是第一个发布的免费软件(即不需要购买就能免费使用)程序。

正巧,也是1975年,年轻的Bill Gates和Paul Allen一起编写了BASIC语言,随后他们把该版本卖给了MIT公司。

20世纪70年代、80年代和90年代,研究进程继续飞速发展,越来越多的发展导致了现在的一种状况:有无数种各有千秋的不同编程语言存在。同时出现了互联网,它是多种语言的集成。互联网另一重要的优势是很容易与别人共享信息和程序,这也就意味着通过自由交换信息、思想和应用程序使得人们对编程和编程语言的兴趣越来越浓厚。

以上简单回顾了编程的发展过程,重点讲述了一些对当今编程领域产生重大影响的事件。接下来介绍编程的概念。

1.2 编程的概念

编程的概念没有统一的定义,不同人有不同的定义。

然而,本书作者对此提出一种定义:

编程是利用某种语言与计算机对话,计算机能理解这种编程语言的文法和语法,然后帮助用户完成有用的工作。

把它翻译成最简单的术语就是编程!用户编写代码,计算机解释用户的请求并执行请求。

完成任务是编程最重要的部分。编写程序就是一直让计算机来执行任务(即使它在等待另一条指令)。编程就是执行一个任务后继续往下执行。无论什么时候,不管是简单的还是复杂的任务,总是给计算机一条指令。通常这些指令是同时给计算机的,尽管有时候似乎让计算机同时执行许多不同的任务,但实际还是给出一步一步的指令让计算机来执行。

代码必须是正确的,不能具有多义性,不能存在任何错误或多义性,否则代码会出错,不能运行。运行代码时,计算机不能猜测代码的意思,不能纠正错误。

另外要注意的是,开始编程时,只执行一项任务的代码不代表只做一个工作,即使最简单的项目通常也由几部分组成,例如:

- (1) 执行程序
- (2) 检查初始化参数
- (3) 改变参数
- (4) 运行后清除一些文件
- (5) 退出程序

1.2.1 存在多种编程语言的原因

如果选择一种计算机可以理解的指令格式，如何在许多不同的计算机语言中选择一种语言呢？如何能确保编写的代码计算机能理解呢？

然而不同类型的计算机理解不同类型的代码(因此 PC 机和 Macintosh 机完全不同)，每一类型的计算机其实只能理解一种语言，这种语言并不是用来编写代码的语言，编写的代码计算机并不能理解。程序(解释程序或编译器)被要求把文本转换成计算机可读的二进制语言。这是计算机语言中的主要部分。

计算机通过读二进制的指令运行，对于二进制，用户可能都知道，它是一个数字系统但不同于通常的十进制，它只有两个数字——0 和 1。图 1-1 显示了 0~9 的十进制和二进制的相互转换。

十进制数字: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9	
二进制位: 0, 1	
十进制	二进制
0	0
1	1
2	10
3	11
4	100
5	101
6	110
7	111
8	1000
9	1001

图 1-1

接着再看图 1-2。

```

01000010011010011011011100110000101110010011110010010000001101001011100110010000
0011101000110010101100100011010010110111101110101011100011001000000110000101101110
01100100001000000110100101100110001000000111011101100101001000000110100001100
0000110100101101110001000000110001001101001011011100110000101110010011100100100
0000011101110110100101110100011010000010000011011101110101011100100010000011
00011011011101101010111000001101010111010001100101011100100111001100100000111
01010111001101101001011011100110011001000001101001011101000010000001100001011
010001101100001000000111010001101000011001010010000001110100011010010110110101
1001010010000001110100011010000110100101110011001110111001100100000011101101
10111101110101011011000110010000100000011001110110010101110100001000000111011001
1000101011100100111100100100000011000110110111011011011100000110110001101001011
0001101100001011101000110010101100100001000000110100101101110011001000110010101
1001010110010000100000001010000110000101101100011101000110100001101111011101010
110011101101000001000000110111101110101011101000010000001101011011001010111100101
1000100110111101100001011100100110010000111001100100000011101110111101110101011
01100011001000010000001100010011001010010000001100001001000000110110001101111
0111010000100000011100110110100100110110111000001101100011001010111001000010000
1001010010010110

```

图 1-2

二进制冗长单调，如果使用计算机一定要用二进制，实际上所有事情都会很复杂(尽管键盘输入会简单很多)。

设想一下，图 1-2 表示什么意思？这是用二进制表示 ASCII 码(用于信息交换的美国标准

代码)的字母的最后一段。不用考虑这些数字表示什么意思(如果愿意的话,以后来回顾这段数字),现在,要感谢编程让我们不需写一串串无止尽的0和1。

编程的时候,确切地说是编写代码的时候,真正要做的是编写另一程序可以理解的代码,这个程序能把代码翻译成计算机可以理解的代码,称为解释程序(interpreter)。因此,编写代码时不是遵循计算机的规则,而是遵循能把代码翻译成计算机可读代码的解释程序的规则(简单地说,就是编译器)。

图 1-3 是代码编写过程的简化图。

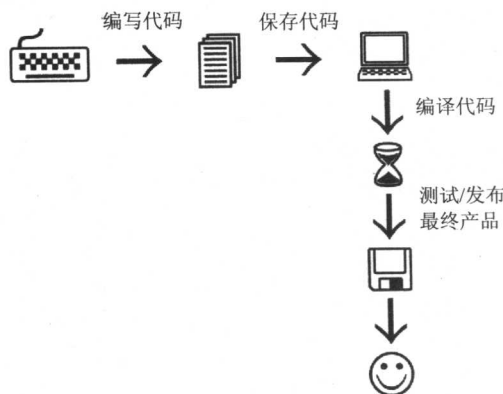


图 1-3

1.2.2 不同的代码, 相同的结果

下面这段用各种语言编写的代码可以说明不同的代码能得到相同的结果, 下面的几种语言可能您之前已听说过。

1. BASIC 语言

```
10 print "Hello World!"
20 goto 10
```

2. Atari BASIC 语言

```
10 REM HELLO.BAS
20 POKE 764,255
30 PRINT "Hello World"
40 IF PEEK(764)=255 THEN GOTO 30
```

3. C 语言

```
#include <stdio.h>

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

4. C++语言

旧版本的 C++ 代码:

```
#include <iostream.h>

main()
{
    for(;;)
    {
        cout << "Hello World! ";
    }
}
```

较新版本的代码:

```
#include <iostream>

int main()
{
    std::cout << "Hello, World!\n";
}
```

5. COBOL 语言

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID. HELLOWORLD.
000300 DATE-WRITTEN. 02/09/04 17:24.
000400* AUTHOR FRED F
000500 ENVIRONMENT DIVISION.
000600 CONFIGURATION SECTION.
000700 SOURCE-COMPUTER. RM-COBOL.
000800 OBJECT-COMPUTER. RM-COBOL.
000900
001000 DATA DIVISION.
001100 FILE SECTION.
001200
100000 PROCEDURE DIVISION.
100100
100200 MAIN-LOGIC SECTION.
100300 BEGIN.
100400 DISPLAY " " LINE 1 POSITION 1 ERASE EOS.
100500 DISPLAY "HELLO, WORLD." LINE 15 POSITION 10.
100600 STOP RUN.
100700 MAIN-LOGIC-EXIT.
100800 EXIT.
```

6. FORTRAN 语言

```
C
C Hello, world.
C
Program Hello
```