



国外经典教材

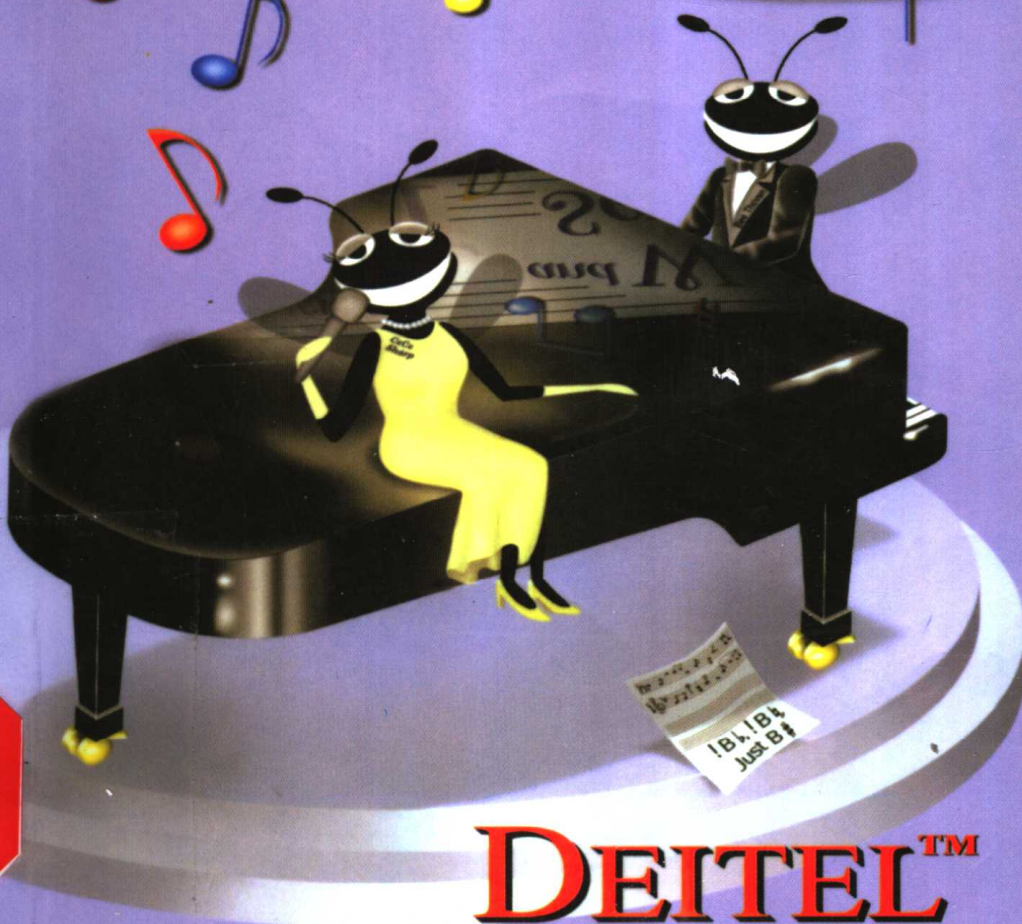
C#大学教程

PEARSON
Prentice
Hall

C# HOW TO PROGRAM

Introducing
NET and Web
Services

- .NET INTRODUCTION
- CONTROL STRUCTURES
- METHODS AND ARRAYS
- CLASSES/DATA ABSTRACTION
- WEB SERVICES/ASP .NET
- COLLECTIONS/PROPERTIES
- DATABASE/ADO .NET/SQL
- OBJECT-ORIENTED PROGRAMMING
- INHERITANCE/POLYMORPHISM
- EXCEPTION HANDLING
- ASSEMBLIES/NAMESPACES
- VISUAL STUDIO .NET IDE
- GUI/WINDOWS FORMS
- WINDOWS (& CUSTOM) CONTROLS
- EVENT HANDLING/DELEGATES
- WEB FORMS/WEB CONTROLS
- XML/XSLT™/DOM™
- INTROS TO SOAP AND BizTALK™
- STRINGS/REGULAR EXPRESSIONS
- MULTIMEDIA/GRAPHICS/GDI+
- NETWORKING/CLIENT-SERVER
- DATA STRUCTURES
- FILES AND STREAMS
- MULTITHREADING
- OPERATOR OVERLOADING
- UNICODE®
- ACCESSIBILITY/VOICEXML™
- COM INTEGRATION
- VISUAL STUDIO .NET DEBUGGER
- DOCUMENTATION GENERATOR
- BIT/CHARACTER MANIPULATION



DEITEL™

DEITEL
DEITEL
LISTFIELD
NIETO
YAEGER
ZLATKINA



(美) H. M. Deitel, P. J. Deitel, J. Listfield, T. R. Nieto, C. Yaeger, M. Zlatkina 著

葛昊吟 汤涌涛 李强 译

清华大学出版社

国外经典教材

C# 大 学 教 程

H. M. Deitel, P. J. Deitel
(美) J. Listfield, T. R. Nieto 著
C. Yaeger, M. Zlatkina

葛昊晗 汤涌涛 李 强 译

清华大学出版社

北 京

内 容 简 介

本书是由世界著名编程语言教材专家组织编写, 讲解了 C# 这种基于 .NET 平台的通用编程语言, 并介绍了如何使用它开发多层结构、客户机/服务器、集中式数据库、基于 Internet 和 Web 的应用程序。内容浅显易懂, 辅之以大量实例, 帮助读者迅速掌握 C#。本书既可作为计算机及相关专业教材, 也可供编程爱好者参阅。

Simplified Chinese edition copyright © 2003 by PEARSON EDUCATION ASIA LIMITED and TSINGHUA UNIVERSITY PRESS.

Original English language title from Proprietor's edition of the Work.

Original English language title: C# How To Program, 1st Edition by H. M. Deitel, P. J. Deitel, J. Listfield, T. R. Nieto, C. Yaeger, M. Zlatkina, Copyright © 2002

EISBN: 0-13-062221-4

All Rights Reserved.

Published by arrangement with the original publisher, Pearson Education, Inc., publishing as Prentice-Hall, Inc.

This edition is authorized for sale only in the People's Republic of China (excluding the Special Administrative Region of Hong Kong and Macao).

本书中文简体翻译版由 Pearson Education 授权给清华大学出版社在中国境内(不包括中国香港、澳门特别行政区)出版发行。

北京市版权局著作权合同登记号 图字: 01-2002-5750

本书封面贴有 Pearson Education(培生教育出版集团)激光防伪标签, 无标签者不得销售。

图书在版编目(CIP)数据

C#大学教程/(美)迪特尔等著; 葛昊晗, 汤涌涛, 李强译.—北京: 清华大学出版社, 2003
(国外经典教材)

书名原文: C# How To Program

ISBN 7-302-07225-6

I. C… II. ①迪… ②葛… ③汤… ④李… III. C 语言—程序设计—高等学校—教材 IV. TP312

中国版本图书馆 CIP 数据核字(2003)第 080079 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

文稿编辑: 姜汉鲁

封面设计: 立日新设计公司

印 装 者: 北京鑫海金澳胶印有限公司

发 行 者: 新华书店总店北京发行所

开 本: 185×260 印 张: 41.5 字 数: 1336 千字

版 次: 2003 年 11 月第 1 版 2003 年 11 月第 1 次印刷

书 号: ISBN 7-302-07225-6/TP·5255

印 数: 1~4000

定 价: 79.00 元

前 言

欢迎来到 C# 世界,并使用 Visual Studio 和 .NET 平台进行 Windows、因特网和万维网编程。本书是新版 .NET How To Program 丛书中的第 2 册,书中给出了 .NET 平台范围内各式各样的前沿计算技术。

C# 是 C 和 C++ 发展的下一个阶段,是特别为微软的 .NET 平台开发的语言。C# 提供了对编程人员来说极其重要的性能,诸如,面向对象编程、字符串、图形、图形用户界面(GUI)组件、异常处理、多线程、多媒体(音频、图像、动画和视频)、文件处理、预封装数据结构、数据库处理、基于因特网和万维网的客户机/服务器网络以及分布式计算等。C# 语言非常适合实现基于因特网和万维网的应用程序,这些应用程序可以与基于 PC 的应用程序无缝连接。

.NET 平台提供了强大的软件开发和部署能力,包括独立于特定的语言 and 平台。与其让开发人员学习新的编程语言,倒不如让他们可以使用任何(或若干)他们所熟知的 .NET 语言(诸如 C#, Visual Basic .NET, Visual C++ .NET 及其他)书写代码,为同一软件项目工作。除了提供语言独立性之外,.NET 通过启用应用程序在多平台上驻留和相互通信,扩展了程序的可移植性,因而使得通过因特网传递 Web 服务更为容易。.NET 平台使得基于 Web 的应用程序分布于消费类电子设备(诸如移动电话和个人数字助理)之中,就像分布在台式机上一样。微软公司并入到 .NET 平台的功能创立了全新的软件开发典范,大大提高了编程人员的生产率并缩短了开发周期。

本书中的新特性

本版包含许多新的特性和增强内容,包括:

- **突出显示**——本书为了更加醒目,将关键元素突出显示,示例如下:

```
comments
keywords
literal values
errors and ASP .NET directives
text, class, method and variable names
```

- **“代码清洗”**——这一术语用于本书中程序格式化的应用过程,使得程序具有细致的注释和开放的布局。考虑到全书包含了近 23 500 行代码,因此采用突出显示并分组为小段的、编档良好的片段,这极大改善了代码的可读性,这也是我们的一个特别重要的目的。

- **Web 服务和 ASP .NET**——微软的 .NET 战略将 Internet 和 Web 包含为一个整体应用到软件开发和部署过程中。Web 服务是这种战略中的一个关键技术,应用标准的 Internet 协议和技术进行信息共享、商业活动和其他交互活动(诸如 HTTP, SOAP 和 XML 等)。Web 服务使得编程人员可以将 Web 转变为可重用软件组件库的形式,用这种形式将应用程序功能打包。在第 21 章中,我们给出了允许用户预订航班的 Web 服务。在这个例子中,用户访问 Web 页面,选定座位选项并将页面提交到 Web 服务器。然后该页面调用 Web 服务检查座位是否可用。集成到 Visual Studio .NET 中的 Crystal Reports 提供了以 Web 服务的形式陈列报告的能力。

- **Web 窗体、Web 控件和 ASP .NET**——应用程序开发人员必须掌握创建基于 Web 的健壮的和可扩展的应用程序。.NET 平台构架支持这样的应用程序。微软的 .NET 服务器端技术——ASP .NET 允许编程人员构建响应客户端请求的 Web 文档。为了启用交互式 Web 页,服务器端程序需要处理用户输入到 HTML 窗体中的信息。ASP .NET 与 ASP 以前的版本有着显著不同,允许开发人员使用 .NET 中强大的面向对象语言编制基于 Web 的应用程序。ASP .NET 同时也提供了增强的可视化编程能力,这种能力与那些为桌面系统构建 Windows 窗体的应用相类似。编程人员可以通过向 Web 窗体上拖放 Web 控件,可视化地创建

Web 页面。在第 20 章中将介绍这些强大的技术。

- **面向对象编程**——面向对象编程是用于开发稳健的、可重用的软件中的最为广泛的技术,而且 C# 提供了增强的面向对象编程技术。书中对面向对象编程做了大量阐述。在第 8 章中介绍了如何创建类和对象。这些概念在第 9 章中进行了扩展,讨论了编程人员如何“吸收”现有类中的功能来创建新类。第 10 章使读者熟悉多态、抽象类、具体类和接口等关键性概念,这对于继承层次结构中的对象来说,可增强其操作能力。

- **XML**——使用 XML 来探索软件开发行业、电子商业和电子商务领域,并且普遍贯穿于 .NET 平台。因为 XML 在描述数据和创建标记语言中是独立于平台的技术。XML 的数据可移植性与 C# 的可移植应用程序与服务完美地集成在一起。在第 18 章中介绍了 XML 语言,其中介绍了基本的 XML 标记,并讨论了用于确认 XML 文档内容的技术,诸如文件类型定义(DTD)和模式(Schema)。同时也解释了如何使用文档对象模型(Document Object Model, DOM)来对 XML 进行编程操作,以及如何通过可扩展样式表语言转换(Extensible Stylesheet Language Transformations, XSLT)将 XML 文档转换成其他类型的文档。

- **多线程**——我们可以在计算机上并行(或同时)执行多个任务,诸如打印文档、从网络下载文件和浏览 Web。通过多线程技术,编程人员可以开发出执行并发任务的应用程序。在历史上,计算机只有单一的、昂贵的处理器供所有的应用程序共享其操作系统。今天,处理器越来越便宜,使得有可能构建出包含多个处理器并行工作的计算机——称其为多处理器计算机。多线程对单处理器系统和多处理器系统都是有效的。C# 的多线程能力使得它的平台及其相关技术,在处理当今复杂的多媒体密集型、数据库密集型、基于网络和基于多处理器的分布式应用程序上有了更充分的准备。第 14 章中提供了有关多线程的详细探讨。

- **ADO .NET**——数据库存储着大量的信息供个人和组织访问和处理业务。随着微软活动数据对象(ActiveX Data Objects, ADO)的发展,ADO .NET 给出了为构建与数据库交互的应用程序,提供了一个新方法。ADO .NET 使用 XML 和一个增强的对象模型,给开发人员提供了他们访问和处理数据库所需的工具,用于大规模、可扩展的、任务关键的多层应用程序。第 19 章中将详细介绍 ADO .NET 的功能和用于操纵数据库的结构化查询语言(SQL)。

给教师的一些提示

学生钟爱学习前沿语言

H. M. Deitel 博士在大学里讲授编程入门课程已有 20 年的历史,在教学中强调开发编写清晰、设计出色的程序。在这些课程中主要讲授了编程的基本原理,集中表现在数据类型、控制结构、数组和函数的有效使用上。依照我们的经验,学生处理书中提及的素材与其处理其他入门和中级编程课程中的内容没什么两样。有一个值得注意的区别就是:学生有高度的学习动力,这种动力来自以下方面,即他们正在学习一种位于前沿的编程语言——C# 和一种位于前沿的编程典范——面向对象编程,这些都将在他们进入业务领域时立竿见影地发挥作用。由于学生必须同时掌握基本语言和实体类库,因而要在 C# 课程中讲述更多的内容,以这些素材作为学习的要点将提高学生对素材的积极性。尽管 C# 是一种新的编程语言,要求编程人员更新其技能,编程人员仍会积极主动地照做,因为微软在 .NET 中提供了大量丰富的、功能强大的功能。

面向对象的世界

在 20 世纪 90 年代末,大学里仍在强调过程化编程。面向对象的 C++ 也曾作为一门前沿课程,但其中大都混有相当数量的过程编程、向 C++ 要求程序员做的是面向对象编程。许多教师如今开始强调纯粹的面向对象编程方式。本书突出了面向对象的编程方式,因为 C# 中提供的就是面向对象。

本书焦点

我们的目标非常明确:面向那些没有或很少编程经验的学生,针对专业人员和学生在传统的高级编程课程中所提出的需求,为大学基础计算机编程课程编写 C# 的教材,并提供对理论与实践的深入探讨和严

格的处理方法。为达到这一目标,我们编写了一本全面细致讲授计算机编程和 C# 语言的原理的书籍,包括控制结构、面向对象编程、C# 类库、图形用户界面的概念、事件驱动的编程等等。在掌握了本书的内容后,学生即完全有能力使用 C# 和在 .NET 上编程。

教学方法

书中收集了丰富的实例、练习和项目,这些项目涉及许多领域,设计它们的目的是提供给学生一些机会,去解决一些有趣的、现实生活中的实际问题。书中代码均在 Windows 2000 和 Windows XP 上测试过。本书侧重于良好软件工程的原则,重点强调程序的明确性。我们作为教育者,在遍及全世界各行各业的教室中讲授一些前沿的课题。我们会通过利于教学的实例来避免晦涩的术语和语法定义。本书侧重于良好的教学方法。

LIVE-CODE 教学法

书中包含大量的 LIVE-CODE 范例。这种风格例证了我们对于程序的教学和编写代码的方式,并将注意力集中到多媒体网络教室和基于 Web 的培训课程上。每个新的概念都采用完全上下文相关的方式给出,每个运行的程序紧接着会有一个或多个窗口,用于显示程序的输入/输出对话框。我们将这种教学方法称为 LIVE-CODE 教学法。我们用编程语言来讲授编程语言。阅读书中的实例倒更像是在计算机上输入和运行这些程序。

万维网访问^①

本书(以及我们其他的发行物)中所有的范例的代码均可以通过以下网站下载:

www.deitel.com

www.prenhall.com/deitel

注册过程快捷易行,且所有代码均免费下载。我们建议读者下载全部范例,然后在阅读相关章节时逐个运行。对程序进行调整,就会马上看到其效果——这就是学习编程的主要方法。每组指令都假定读者是在 Windows 2000 或 Windows XP 上运行,并使用了微软 Internet Information Services (IIS) 服务。另外,在网站还能找到 Web 服务器的设置指令和其他一些软件。

包含 C# 的 Visual Studio .NET 可以从微软订购和下载。Visual Studio .NET 共有 3 个版本,分别是企业版、专业版和学术版。详情请访问网站 developerstore.com/devstore/。如果是 Microsoft Developer Network (MDN) 成员,则可以访问网站 msdn.microsoft.com/default.asp。

目标

在每章开头首先提出期待学生达到的目标,以及阅读这一章之后确定他们是否已经达到预定的目标。目标的作用就是给学生树立信心和作为积极地强化教学的动力。

书中包含了 204 个范例程序,将近 23 500 行代码

我们借助于完整的、可运行的 C# 程序介绍 C# 的特性。程序的规模不一,小到几行代码、大到包含数百行代码。所有的代码均可以到我们的网站 www.deitel.com 下载。

607 个图表

书中包括丰富的图表、线条图和程序输出。例如,在讨论控制结构时,用流程图详细地描绘了其各种性能。注意:这里不会将流程图作为程序开发工具进行讲授,但我们确实使用一种简明的流程图来解释每个

^① 原书部分内容中译版请参阅:<http://www.34.cn>——译者注。

C# 控制结构的精确操作。

509 条编程提示

书中包含的程序提示可以帮助学生关注程序开发的重要方面。从良好编程习惯、常见编程错误、测试与调试技巧、性能提示、可移植性提示、软件工程知识和界面常识等方面来强调这些提示。

这些提示和习惯是作者多年的编程和教学经验的结晶。有一位数学专业的学员告诉我们,她感觉这种方式就像数学书中的公理、定理和推论的强调方式一样,为构建良好的软件奠定了基础。

91 条“良好编程习惯”

良好编程习惯作为提示,目的是让学生关注有助于他们编制良好的程序所需技术的注意。当我们给非编程人员讲授入门课时,我们会“明确”每一门课的“专业术语”,还会告诉学生:我们将重点强调编写更有条理、更加易懂和更易维护的程序时所采用的技术。

165 条“常见编程错误”

学生在学习一种语言,特别是第一次接触编程课程时,通常会频繁地犯特定种类的错误。指出这些常见编程错误,会减少学生犯同样错误的机会,同时也会缩短办公时间和教师答疑的时间。

44 条“测试和调试技巧”

最初设计这种“提示类型”,我们认为提示中应当包括对所暴露的错误以及将其从程序中移除的严格建议。事实上,许多提示都表明 C# 是将排除错误放在第一位的,从而简化了测试和调试过程。

57 条性能提示

依照我们的经验,到目前为止,教会学生写出清晰易懂的程序是第一门编程课程中最重要的目标。但是学生想通过其他方式要编写出运行得最快、耗用内存最少、击键次数最少或迷惑最少的程序。学生的确关心性能,并且他们也想知道如何优化他们的程序。本书包括 57 条性能提示,重点突出改善程序性能的时机——使程序运行更快或减少所占有的内存。

16 条“可移植性提示”

可移植性提示可以帮助学生编写可移植代码,并提供了 C# 如何实现其高度可移植性的常识。

115 条“软件工程知识”

面向对象编程范例需要对构建软件系统的方式重新进行全面地规划。C# 是实现良好软件工程的有效语言。软件工程知识重点突出了影响软件系统构成的设计思路与体系结构,尤其针对大型系统。学生在此所学到的知识对高级课程是非常有用的,同时对他们在工作中开始着手的大型的、复杂的实际系统也有所帮助。

21 条“界面常识”

书中使用界面常识来强调图形用户界面的约定,帮助学生设计吸引人的、友好的图形用户界面,这些界面应该遵循行业规范。

小结(共 1277 项)

在每章后都有附加的教学计划。以项目列表的形式给出了整章的全面总结。平均每章有 39 条。帮助学生复习和加强关键概念的理解。

693 道自测题和答案

为便于自学,本书提供了大量的自测题和答案。这些问题和答案给学生提供了一个机会,去使用这些素材建立自信,并为普通练习做好准备。应该鼓励学生去尝试所有的自测题,然后核对答案。

367 道练习题

每章都以充实的练习题结束,包括对重要术语和概念的简单回顾,编写单独 C# 语句,编写 C# 方法和类中的一段程序,编写完整的 C# 方法、类和应用程序,编写专业项目等等。这些练习广泛涵盖了各类主题,使得教师可以将其课程按照受众的独特需求和每学期的不同课程安排来进行设计。教师可以将这些练

习作为家庭作业、小测验和专业考试使用。

作者简介

H. M. Deitel 博士: Deitel & Associates 公司主席兼首席战略官, 在计算领域已有 41 年的工作经验, 无论专业技术还是学校教育, 均有非常高的造诣。H. M. Deitel 博士拥有麻省理工学院的 B.S 和 M.S 学位, 以及波士顿大学的 Ph.D 学位。他参与过 IBM 和 MIT 的一系列领先于时代的虚拟内存操作系统项目, 研究成果如今已广泛用于 UNIX, Windows NT, OS/2 和 Linux 系统。他有 20 年的大学教学经验, 而且在其子 P. J. Deitel 成立 Deitel & Associates 公司之前, 一直担任波士顿大学计算机科学系主任的职位。他创作或参与创作了数十本书, 并参与了多媒体产品的开发。多年来, 他的作品已被翻译成简体中文、繁体中文、日语、俄语、西班牙语、朝鲜语、法语、波兰语、意大利语、葡萄牙语以及希腊语, 畅销全球。H. M. Deitel 博士经常举办国际性技术研讨会, 向大型公司、政府部门以及各种军事单位“传经布道”。

P. J. Deitel: Deitel & Associates 公司 CEO 兼首席技术官, 毕业于麻省理工学院的斯隆管理学院, 主修信息技术。在 Deitel & Associates 公司, 他负责面向业内许多知名客户讲授 Java, C, C++, Internet/万维网课程, 这些客户包括康柏、Sun、White Sands Missile Range、Rogue Wave Software、Computervision、Stratus、Fidelity、Cambridge Technology Partners、Lucent Technologies、AdraSystems、Entergy、CableData Systems、NASA 肯尼迪太空中心、美国国家大风暴实验室(NSSL)、IBM 以及其他许多公司和机构。他负责为计算机机构联盟波士顿分部讲授 C++ 和 Java 课程。他目前正用 Deitel & Associates、Prentice Hall 以及“美国技术教育网络”(Technology Education Network) 联合投资的一笔风险基金, 利用卫星技术来提供 Java 课程。他和其父 H. M. Deitel 博士是全球畅销程序语言教科书的作者。

T. R. Nieto: Deitel & Associates 公司产品开发主管, 毕业于麻省理工学院, 主修工程与计算。在 Deitel & Associates 公司, 他负责向业内的一系列客户发布培训课程, 这些客户包括 Sun、康柏、EMC、Stratus、Fidelity、NASDAQ、Art Technology、Progress Software、Toys “R” Us、美国国家海洋和天气管理局的运营支持部、喷气推进实验室、Nynex、摩托罗拉、芝加哥联邦储备银行、Banyan、Schlumberger、Notre Dame 大学、NASA、惠普、大量军事基地和其他许多单位。他和两位 Deitel 先生合作出版了多种计算机书籍和多媒体教学光盘。事实上, Deitel & Associates 出版的几乎每一样产品都有他的功劳。

C. H. Yaeger: Deitel & Associates 公司的 Microsoft 软件出版物总监, 毕业于波士顿大学, 3 年内就获得了计算机科学学士学位。Cheryl 参与创作了 Deitel & Associates 的大量出版物, 其中包括《C# How to Program》、《C# A Programmer's Introduction》、《C# for Experienced Programmers》和《Visual Basic .NET for Experienced Programmers》。另外, 他还为 Deitel 其他出版物写过稿, 包括《Perl How to Program》、《Wireless Internet & Mobile Business How to Program》、《Internet and World Wide Web How to Program (第二版)》和《Visual Basic .NET How to Program (第二版)》。

M. Zlatkina: 毕业于布兰代斯大学, 3 年内就获得计算机科学和数学学士学位, 第 4 年获得计算机科学硕士学位。在布兰代斯大学上学期间, 她领导了数据库领域的研究, 并成为一名教学助理。她参与创作了《C# How to Program》、《C# A Programmer's Introduction》和《C# for Experienced Programmers》, 并为另一本 Deitel 出版物写过稿, 即《e-Business & e-Commerce for Managers》。

J. A. Listfield: 哈佛大学应届毕业生, 主修计算机科学。他的毕业论文涉及了计算机图形、网络 and 计算理论, 并有 C, C++, Java, Perl 和 Lisp 的编程经验。Jeff 参与创作了《C# How to Program》、《C# A Programmer's Introduction》和《C# for Experienced Programmers》, 并参与创作了《Perl How to Program》。

万维网联盟(World Wide Web Consortium, W3C)

Deitel & Associates 是万维网联盟的成员, 该联盟成立于 1994 年, 为万维网的不断发展开发公共协议。作为万维网联盟成员, Deitel & Associates 公司在 W3C 顾问委员会中占有一席之地(目前任职的是技术总监 Paul Deitel)。顾问委员会通过在世界范围内举行会议, 协助为 W3C 提供战略性方向。各成员机构同时



通过参与 W3C 活动和小组,协助为 Web 技术(诸如 XHTML、XML 等其他)开发标准化建议。W3C 中各成员均为公司和大型组织。要获取关于成为 W3C 成员的更多信息请访问以下网站:

www.w3.org/Consortium/Prospectus/Joining.

目 录

第 1 章 计算机、因特网、万维网和C#	1	3.3 另一个简单的程序:整数求和	45
1.1 概述	1	3.4 内存的概念	48
1.2 什么是计算机	1	3.5 算术运算	49
1.3 计算机结构	2	3.6 判断:相等和关系运算符	51
1.4 操作系统的演化	2	小结	55
1.5 单机计算、分布式计算和客户机/ 服务器计算	3	自测题	57
1.6 机器语言、汇编语言和高级语言	3	自测题答案	57
1.7 C,C++,Visual Basic .NET 和 Java	4	练习题	58
1.8 C#	5	第 4 章 控制结构(一)	61
1.9 其他高级语言	6	4.1 概述	61
1.10 结构化编程	6	4.2 算法	61
1.11 重要的软件趋势:对象技术	6	4.3 伪代码	61
1.12 硬件趋势	7	4.4 控制结构	62
1.13 因特网和万维网的历史	8	4.5 if 选择结构	63
1.14 万维网联盟(W3C)	9	4.6 if/else 选择结构	64
1.15 可扩展标记语言(XML)	9	4.7 while 循环结构	67
1.16 .NET介绍	10	4.8 算法描述:案例研究一(计数控制循环)	68
1.17 .NET框架和公共语言运行库	11	4.9 算法描述:案例研究二(标记控制循环)	70
1.18 本书导读	12	4.10 算法描述:案例研究三(嵌套控制结构)	75
小结	17	4.11 赋值运算符	78
自测题	17	4.12 递增和递减运算符	79
自测题答案	18	4.13 Windows 应用程序介绍	81
练习题	18	小结	86
第 2 章 Visual Studio .NET集成开发环境	19	自测题	88
2.1 概述	19	自测题答案	89
2.2 Visual Studio .NET集成开发环境简述	19	练习题	90
2.3 菜单栏和工具栏	21	第 5 章 控制结构(二)	93
2.4 Visual Studio .NET窗口	22	5.1 概述	93
2.5 使用【帮助】	25	5.2 计数控制循环的实质	93
2.6 简单的程序:显示文本和图像	26	5.3 for 循环结构	94
小结	33	5.4 for 结构应用举例	97
自测题	35	5.5 switch 多选结构	101
自测题答案	35	5.6 do/while 循环结构	104
练习题	35	5.7 break 和 continue 语句	106
第 3 章 C# 编程介绍	37	5.8 逻辑和条件运算符	108
3.1 概述	37	5.9 结构化编程总结	112
3.2 简单的程序:打印一行文本	37	小结	116

自测题	117	8.3 类的作用域	204
自测题答案	118	8.4 类成员的访问控制	204
练习题	119	8.5 初始化类对象:构造函数	205
第 6 章 方法	120	8.6 使用重载构造函数	206
6.1 概述	120	8.7 属性	209
6.2 C# 中的程序模块	120	8.8 合成:对象引用作为其他类的实例变量	216
6.3 Math 类方法	121	8.9 使用 this 引用	219
6.4 方法	122	8.10 垃圾回收	220
6.5 方法的定义	123	8.11 静态类成员	221
6.6 参数提升(转换)	130	8.12 常量和只读成员	224
6.7 C# 命名空间	132	8.13 索引器	226
6.8 数值类型和引用类型	132	8.14 数据抽象和信息隐藏	231
6.9 传输参数:值传递与引用传递	133	8.15 软件重用	232
6.10 生成随机数	136	8.16 命名空间和程序集	233
6.11 例子:机率游戏	142	8.17 【类视图】和【对象浏览器】	237
6.12 变量的持续周期	145	小结	239
6.13 变量的作用范围规则	146	自测题	240
6.14 递归	148	自测题答案	241
6.15 使用递归的实例	151	练习题	241
6.16 递归与迭代的比较	153	第 9 章 面向对象编程:继承	243
6.17 方法重载	153	9.1 概述	243
小结	155	9.2 基类和派生类	244
自测题	157	9.3 受保护成员和内部成员	245
自测题答案	159	9.4 基类和派生类的关系	246
练习题	161	9.5 案例研究:三级继承层次结构	261
第 7 章 数组	164	9.6 派生类中的构造函数和析构函数	264
7.1 概述	164	9.7 与继承相关的软件工程	269
7.2 数组	164	小结	269
7.3 声明和分配数组	166	自测题	270
7.4 数组应用举例	166	自测题答案	271
7.5 向方法传递数组	174	练习题	271
7.6 通过值和引用传递数组	176	第 10 章 面向对象编程:多态性	272
7.7 数组的排序	180	10.1 概述	272
7.8 查找数组:线性查找法和折半查找法	182	10.2 派生类对象转换为基类对象	272
7.9 多维数组	186	10.3 类型字段和 switch 语句	277
7.10 foreach 循环结构	192	10.4 多态性示例	277
小结	192	10.5 抽象类和方法	278
自测题	193	10.6 示例(一):接口和实现的继承	279
自测题答案	194	10.7 密封的类和方法	286
练习题	194	10.8 示例(二):使用多态性的工资发放系统	286
第 8 章 基于对象的编程	198	10.9 示例(三):创建和使用接口	294
8.1 概述	198	10.10 委托	304
8.2 用类实现一个 Time 抽象数据类型	199	10.11 运算符重载	307

小结	312	小结	422
自测题	312	自测题	424
自测题答案	313	自测题答案	424
练习题	313	练习题	425
第 11 章 异常处理	314	第 14 章 多线程	426
11.1 概述	314	14.1 概述	426
11.2 异常处理概述	314	14.2 线程状态:线程的生命周期	427
11.3 示例:DivideByZeroException	316	14.3 线程的优先级和线程调度	428
11.4 .NET的异常层次结构	320	14.4 线程同步和类监视器	432
11.5 finally 块	321	14.5 没有线程同步的生产者/消费者关系	433
11.6 Exception 属性	326	14.6 有线程同步的生产者/消费者关系	438
11.7 程序员自定义的异常类	330	14.7 生产者/消费者关系:循环缓冲区	444
11.8 用 checked 和 unchecked 运算符 处理溢出	333	小结	453
小结	335	自测题	454
自测题	337	自测题答案	455
自测题答案	338	练习题	455
练习题	338	第 15 章 字符、字符串和正则表达式	457
第 12 章 图形用户界面概念(一)	339	15.1 概述	457
12.1 概述	339	15.2 字符和字符串基础	457
12.2 Windows 窗体	340	15.3 String 类构造函数	458
12.3 事件处理模型	342	15.4 String 类的索引器、Length 属性和 CopyTo 方法	459
12.4 控件属性和布局	346	15.5 字符串比较	460
12.5 标签、文本框和按钮	349	15.6 String 类的 GetHashCode 方法	463
12.6 分组框和面板	354	15.7 字符串定位查找	464
12.7 复选框和单选钮	356	15.8 String 类的子串提取	467
12.8 图片框	363	15.9 字符串相加	468
12.9 鼠标事件处理	365	15.10 String 类中的其他方法	468
12.10 键盘事件处理	367	15.11 StringBuilder 类	470
小结	369	15.12 StringBuilder 索引器、Length 和 Capacity 属性以及 EnsureCapacity 方法	471
自测题	370	15.13 StringBuilder 类的 Append 和 AppendFormat 方法	473
自测题答案	371	15.14 StringBuilder 类的 Insert, Remove 和 Replace 方法	475
练习题	371	15.15 Char 方法	478
第 13 章 图形用户界面概念(二)	373	15.16 模拟扑克洗牌和发牌程序	480
13.1 概述	373	15.17 正则表达式和 Regex 类	484
13.2 菜单	373	小结	492
13.3 LinkLabel 控件	380	自测题	494
13.4 列表框和复选列表框	383	自测题答案	495
13.5 组合框	389	练习题	495
13.6 TreeView 控件	393		
13.7 ListView 控件	397		
13.8 选项卡控件	403		
13.9 多文档界面(MDI)窗口	408		
13.10 可视化继承	415		
13.11 用户自定义控件	418		

第 16 章 图形和多媒体	496	17.8 创建随机访问文件	583
16.1 概述	496	17.9 在随机访问文件中随机写入数据	586
16.2 图形上下文和图形对象	497	17.10 从随机访问文件中顺序读取数据	590
16.3 颜色处理	498	17.11 案例研究:交易处理程序	594
16.4 字体处理	504	小结	611
16.5 绘制线段、矩形和椭圆	508	自测题	612
16.6 绘制弧线	511	自测题答案	613
16.7 绘制多边形和折线	513	练习题	613
16.8 高级图形功能	516	第 18 章 数据库、SQL 和 ADO.NET	615
16.9 多媒体概述	521	18.1 概述	615
16.10 载入、显示和缩放图像	521	18.2 关系数据库模型	615
16.11 连续图像动画	523	18.3 关系数据库概述:Books 数据库	616
16.12 Windows 媒体播放器	534	18.4 结构化查询语言(SQL)	620
16.13 Microsoft Agent	536	18.5 ADO.NET 对象模型	631
小结	548	18.6 使用 ADO.NET 编程:从 DBMS 中提取信息	631
自测题	549	18.7 使用 ADO.NET 编程:修改 DBMS	639
自测题答案	549	18.8 读写 XML 文件	645
练习题	549	小结	647
第 17 章 文件和流	551	自测题	649
17.1 概述	551	自测题答案	650
17.2 数据层次结构	551	练习题	650
17.3 文件和流	553		
17.4 File 和 Directory 类	553		
17.5 创建顺序访问文件	561		
17.6 从顺序访问文件中读取数据	571		
17.7 随机访问文件	580		

第 1 章 计算机、因特网、万维网和C#

学习目标

- 理解基本的计算机概念
- 学习各种编程语言
- 熟悉C#编程语言的历史
- 理解微软在.NET上的创新
- 预览本书的其他章节

1.1 概述

欢迎来到C#的世界！编写本书的过程中，我们努力地工作，以期向读者提供最准确和最完整的有关C#语言以及.NET平台的信息。本书适合各个层次的读者，无论是几乎没有任何编程经验的初学者还是熟练的程序员。我们希望对于读者来说，本书的学习是一个丰富知识、充满乐趣而又富于挑战的过程。

如何才能使一本书既适合初学者又适合有经验的程序员呢？本书的核心在于：通过使用结构化编程、基于对象编程、面向对象编程(OOP)和事件驱动编程这些成熟的技术，来强调清晰化编程的实现。初学者所学习的基本技能是为良好地编程打基础；有经验的开发人员则可以深入领会编程语言的严格定义，改进他们的编程风格。也许更重要的是本书展示了数百个完整的C#程序及其输出结果。这称为“活代码”(LIVE CODE)方法。本书中的所有例子都可以通过我们的网站 www.deitel.com 获得。

几乎在每个不断发展的领域中，计算机的应用都在不断增加。虽然处于成本稳步上升的时代，计算的成本却在戏剧性地下降，这得益于硬件和软件技术的飞速发展。20多年前，占满了整个大屋子，价值几百万美元的计算机，其CPU现在已经能在一块比指甲还小的硅芯片表面上制造出来，并且只耗费很少成本。硅是地球上最丰富的物质之一，只是普通沙子的一个成分。硅芯片技术已经使计算变得如此经济，以至于世界范围内有上亿台普通计算机投入使用，遍及商业、工业、政府和个人生活的各个领域。鉴于目前技术的发展速度，其数量很快会在未来几年内翻一番。

开始学习本书，也就是开始了一段富有挑战和收获的学习历程。随着学习的深入，如果遇到问题，请发电子邮件到 deitel@deitel.com 与我们联系，或者浏览我们的网站 www.deitel.com, www.prenhall.com/deitel 以及 www.informit.com/deitel。我们希望读者能通过阅读本书获得学习C#的乐趣。

1.2 什么是计算机

计算机是一种能执行计算和做出逻辑判断的设备，它的计算速度要比人快上百万倍甚至几十亿倍。例如，今天很多的个人计算机能够在一秒内执行上亿次甚至几十亿次加法运算。一台功能强大的计算机一秒内完成的计算，一个人利用台式计算器也许需要几十年才能完成。（需要考虑的是：如何才能知道人所相加的数字是否正确？又如何才能知道计算机对数字进行的相加是否正确？）今天最快的超级计算机每秒能进行上千亿次加法运算，大约是几十万人一年的计算量。每秒万亿次的计算机也已经开始在研究实验室里运行了。

计算机在称为计算机程序的一套指令的控制下处理数据。这些程序通过一系列有序的行为来指导计算机，而这些行为由作为计算机程序员的个人指定。

一台计算机由各种硬件设备组成（例如键盘、屏幕、鼠标、磁盘、内存、CD-ROM 和处理单元）。运行在计算机上的程序称为软件。近年来硬件的成本呈现出戏剧性的下降趋势，这说明个人计算机已经成为一种

消费品。但是,由于程序员在开发功能更加强大且结构更加复杂的应用程序的同时,软件开发的技术并没有明显地改善,因此软件开发的费用却是稳步上升的。本书中将介绍成熟的软件开发方法——自顶向下逐步优化、功能化和面向对象编程,这些方法能减少软件开发的费用。大家普遍认为面向对象编程是一个重大的突破,能极大地提高程序员效率。

1.3 计算机结构

尽管物理外观上不同,实际上每台计算机都可以认为由 6 个逻辑单元或部分组成:

(1)输入单元:是计算机从各种输入设备获得信息(数据和计算机程序)的“接收”部分。然后输入单元把信息交由其他单元进行信息的处理。现在,大部分用户通过键盘和鼠标设备输入信息。其他的输入设备包括麦克风(用来向计算机输入语音)、扫描仪(用来扫描图像)和数码相机(用来照相和制作视频)。

(2)输出单元:计算机的“运送”部分,将计算机处理过的信息交给各种输出设备,在计算机之外便可以利用信息。计算机可以通过各种方式输出信息,包括在屏幕上显示输出,在音频/视频设备上播放输出,将输出打印在纸上,或者利用输出来控制其他的设备。

(3)存储单元:是计算机中可以高速访问、但容量又相对较小的“仓库”,用于数据临时存储。存储单元存有(retain)输入单元输入的信息,使信息能够被迅速有效地处理。除此之外,存储单元还存有处理过的信息,直到将这些信息传送到输出设备。通常,把存储单元称为内存或者主存——随机访问内存(RAM)就是主存的一个例子。主存通常具有易失性,这意味着计算机关机时主存中的数据将丢失。

(4)算术和逻辑单元(ALU):算术和逻辑单元是计算机的“制造”部分。它负责加、减、乘、除等计算的执行。它也包含决策(decision)机制,允许计算机执行决策任务,例如判别(determine)存储在内存中的两项数据是否相等。

(5)中央处理器(CPU):CPU 是计算机中的“管理者”部分,是计算机中的协调员,负责监督计算机其他部分的执行。当存储单元需要读入信息时,CPU 会提醒输入单元,CPU 还会通知 ALU 在计算中何时需要用到存储单元中的数据,并在存储单元向特定的输出设备发送信息时告知输出单元。

(6)二级存储单元:是计算机中可长期保存数据且大容量的“仓库”。二级存储设备,例如硬盘和磁盘,通常保存其他单元不常用到的程序或者数据;在需要这些信息时——也许是几小时、几天、几个月甚至是几年后,计算机就能从二级存储单元获得这些信息。访问二级存储单元需要的时间比访问主存需要的时间长。但是,二级存储单元的每单位价格远远低于主存每单位的价格。二级存储单元是非易失性的,这意味着即使计算机关机了,二级存储单元仍然存储着信息。

1.4 操作系统的演化

早期的计算机一次只能执行一项工作或任务。在这种计算机的操作模式下(通常称为单用户批处理),计算机每次运行一项程序,处理成组的数据(称为批处理)。这些早期系统的用户,一般使用成叠的穿孔卡片向计算机中心提交他们的工作。通常,几小时甚至几天之后,计算的结果才能返回给用户。

为了使计算机的使用更加方便,人们开发出称为“操作系统”的软件系统。早期的操作系统监视和管理计算机各项工作之间的相互转换。为了使计算机操作者切换工作时所花时间更少,操作系统提高了计算机在一段给定时间内能够处理的工作总量,或称为吞吐量。

随着计算机功能变得越来越强大,单用户批处理模式显得效率低下,因为计算机需要花费很多时间来等待缓慢的输入/输出设备完成它们的任务。开发人员注意到了多程序技术,它允许多个任务“共享”计算机的资源,从而获得更好的利用率。多程序涉及到一台计算机上许多工作的“同步”进行,计算机要将资源分给各个工作。然而,早期多程序操作系统的用户仍然依赖于穿孔卡提交工作,然后等待几小时或几天才获得结果。

在 20 世纪 60 年代,几个工业和大学团体倡导分时操作系统。分时是一种特殊的多程序技术,它允许用户通过“终端”(带有键盘和屏幕的设备)访问一台计算机。几十人甚至几百人都能同时使用一台分时计

算机。需要注意的是,计算机实际上并非同时运行所有人的请求。相反,它每次执行一个用户任务的一小部分,然后就转向为下一个用户服务。由于计算机的运行速度非常快,每秒就能给每个用户提供好几次服务。从表面上看好象是多个用户的程序在同时运行。与先前的计算机系统相比,分时系统有了很大进步,因为用户请求能够很快地得到响应,而不是需要等待很长时间获得结果。

目前广泛应用在高级计算中的 UNIX 操作系统,就是源于一个实验性的分时操作系统。Dennis Ritchie 和 Ken Thompson 于 20 世纪 60 年代后期在贝尔实验室开发了 UNIX 系统和 C 语言,UNIX 系统就是用 C 语言写的。他们将 UNIX 的源码自由地分发给任何想使用、修改或者扩展 UNIX 系统的程序员。一个巨大的 UNIX 用户团体迅速发展起来。而随着 UNIX 用户贡献出他们自己的程序和工具,UNIX 系统也不断得以发展。通过无数研究者和开发人员的共同努力,UNIX 已成为一种功能强大而灵活的操作系统,几乎能够处理用户要求的任何类型任务。而 UNIX 也发展了许多版本,包括今天极其流行的开放源码的 Linux 操作系统。一般来说,开放源码产品的源码可以从因特网免费获取。这样开发人员能学习、验证和修改这些源码。通常开放源码的产品要求开发人员公开他们所做的任何改进,以便于开放源码团体能够继续改进这些产品。

1.5 单机计算、分布式计算和客户机/服务器计算

1977 年,苹果计算机掀起了个人计算机的高潮。起初,这只是一个业余爱好者的梦想,但是,随着计算机价格的迅速降低,很多人由于个人或商业用途购买了计算机。1981 年,世界最大的计算机供应商 IBM 推出了 IBM 个人计算机。从此,个人计算迅速在商业、工业和政府部门中得到广泛应用。

最初由苹果和 IBM 倡导的计算机是“单独”的单元——人们在他们自己的机器上工作,然后通过来回传递磁盘来共享信息(这个过程通常被称为“sneakernet”)。尽管早期个人计算机的功能尚未强大到满足分时给多个用户,但是可以将计算机连接成计算机网络,可以通过电话线或者在一个组织内通过局域网(Local Area Network, LAN)实现。这种网络导致了分布式计算的出现,通过分布式计算,一个组织的计算可以通过网络分布到组织内部的各个计算机上,而不只由一台中央处理机执行计算。个人计算机有足够能力来处理个人用户的计算要求和计算机间以电子形式进行信息传输的基本任务。N 层应用程序将应用程序分给许多分布式的计算机,例如,一个 3 层应用程序可以在一台计算机上设置用户接口,在第二台计算机上处理商业逻辑,而数据库位于第三台计算机上。随着应用程序的运行,所有计算机都可以进行交互。

今天最先进的个人计算机功能已经能和 20 年前的数百万美元的计算机相媲美。高性能的台式机(即工作站),给个人用户提供了强大的功能。通过计算机网络,共享信息变得非常容易。在网络中,那些存储程序和数据的计算机称为“服务器”,而分布在网络中的,使用程序和数据的计算机称为“客户机”。这种配置引出了“客户机/服务器计算”(client/server computing)。今天流行的操作系统,例如 UNIX, Solaris, MacOS, Windows2000, WindowsXP 和 Linux 都提供了这种功能。

1.6 机器语言、汇编语言和高级语言

程序员可以用各种语言编写指令,有的指令计算机可以直接理解,而另一些需要中间的“翻译”步骤。尽管现在有数百种计算机语言在使用,但这些不同的语言可以分成 3 大类:

- (1) 机器语言;
- (2) 汇编语言;
- (3) 高级语言。

计算机都只能直接理解自己的“机器语言”。作为每种特定计算机的“天生的语言”,机器语言在计算机硬件设计时定义。机器语言通常由数字串组成(最终简化为 0 和 1),这些数字指示计算机如何去执行它们最基本的操作。机器语言是依赖于机器的,这意味着一种特定的机器语言只能用在一种类型的计算机上。下面这是一段机器语言代码,功能是将“加班工资”和“基本工资”相加,并将结果存入“工资总额”,从中可以看出机器语言对人来说是难以理解的。

```
+1300042774
+1400593419
+1200274027
```

随着计算机的普及,机器语言编程愈发显得效率低下、并且枯燥和易于出错。程序员开始使用类似英语的缩写来代表计算机的基本操作,而不是使用计算机可以直接理解的数字串。这些缩写构成了“汇编语言”的基础。一种“翻译程序”(称为汇编器)以计算机的速度将汇编语言程序转化为机器语言。下面是一段汇编语言代码,也是将“加班工资”和“基本工资”相加,并将结果存入“工资总额”,但显然这些代码比机器语言更易于理解。

```
LOAD      BASEPAY
ADD        OVERPAY
STORE     GROSSPAY
```

这样的代码虽然更利于人的理解,但是计算机却不能理解,除非将其转化为机器语言。

尽管随着汇编语言的出现,计算机的使用迅速增加,但是即使要完成一件最简单的任务,汇编语言仍然需要很多条指令。为了加快编程的过程,产生了高级语言,可以用简单的语句来完成大量的任务。称为“编译器”的翻译程序将高级语言转化为机器语言。程序员使用高级语言写出的指令和日常英语非常相似,并且还包含常见的数学符号。一个用高级语言写的工资程序也许就包含如下的一条语句:

```
grosspay = basepay + overtimepay
```

显然,与机器语言或汇编语言相比,高级语言更受程序员欢迎。

将高级语言程序编译成机器语言将会消耗大量的时间。但是,“解释”程序的开发解决了这个问题,解释程序能绕过编译这一步,直接执行高级语言程序。虽然编译执行的程序比解释执行的程序快,但是解释器在程序开发环境中还是很流行的。在这些环境下,当开发人员增加新功能或者修改错误时,会频繁地修改代码。一旦程序开发完毕,就可以生成一个编译版本使得程序的运行效率最大化。

1.7 C,C++,Visual Basic .NET 和 Java

随着高级语言的发展,新的语言在先前语言的基础上不断出现,C++从C发展而来,C又是从BCPL和B发展而来的。Martin Richards在1967年为了编写操作系统、软件和编译器开发了BCPL语言。在BCPL之后,Ken Thompson发展了B语言。在1970年,Ken Thompson用B语言编写了早期的UNIX操作系统版本。BCPL和B语言都是“无类型”语言,这意味着每个数据项在内存中都占用一个“字”(两个字节)的大小。使用这些语言,程序员可以将每一数据项视为一个整数或者实数。

C语言最早实现于1973年,是由Dennis Ritchie在贝尔实验室从B语言发展而来的。C语言不仅利用了许多BCPL和B语言的重要概念,而且它还提供了数据类型和其他特性。作为UNIX操作系统的开发语言,C语言获得了广泛的认同。如今C语言适用于绝大部分计算机,当今许多的主流操作系统都是用C和C++编写的。C语言是一种独立于硬件的语言,如果精心设计,可以编写出适用于绝大多数计算机的C语言程序。

C++由Bjarne Stroustrup于20世纪80年代早期在贝尔实验室开发成功,它利用Simula 67(一种模拟编程语言)语言中的成分,对C语言进行扩展。C++提供了许多让C语言更加整洁的特性,但更重要的是,C++提供了面向对象编程(OOP)的能力。

每次对功能更加新颖而且更加强大的软件的需求高涨时,快速、准确和经济地编写软件却总是一个很难以实现的目标。然而,通过使用对象或者可重用组件将现实生活中的事物模型化(参见1.11节),能部分地缓解这个问题。软件开发人员发现,用一种模块化的、面向对象的方法来设计和实现,能使软件开发比先前的软件开发技术(如结构化编程)效率更高。而且,面向对象编程通常也更易于理解、修改和完善。

除了C++,还出现了其他的面向对象的语言。其中包括Smalltalk,它由施乐的Palo Alto研究中心(PARC)发明。Smalltalk是一种纯粹的面向对象语言,顾名思义,每件事都是对象。C++是一种混合语言,它可以编写出类似C语言风格的程序,也可以是面向对象风格的,或者两者兼有。尽管有人认为这样的选