高等学校**应用型特色**规划教材

C++面向对象程序设计

——基于Visual C++ 2010

吴克力　编著

赠送电子教案

- 以面向对象技术为核心，循序渐进，每一章精心设置“案例实训”，强化编程技能的培养
- 利用调试跟踪工具剖析关键知识点，化抽象为直观，强化基本概念的掌握
- 设计基于C++/CLI的窗体应用程序，与时俱进，强化语言实际应用的能力
- 配备丰富的免费教学资源——电子课件与案例实训资源包

清华大学出版社

高等学校应用型特色规划教材

C++面向对象程序设计——基于Visual C++ 2010

吴克力　编著

清华大学出版社
北京

内 容 简 介

本书以面向对象技术为核心，重点介绍了标准C++的语法规则和编程技术。为便于深入理解C++的基本概念和实现技术，书中利用程序调试工具深入浅出地剖析了重要的语法现象和程序运行机理，使初学者能知其然，更知其所以然。书中用两章的篇幅分别介绍了C++/CLI应用程序和WinForm窗体应用程序的设计方法，以便拓展学习者用C++开发应用项目的能力。全书通过丰富的例程、案例和练习培养并锻炼读者的编程能力，使读者能尽快掌握面向对象编程思想和提高编程的技能。

本书既注意对基本概念、基础知识的讲解和剖析，更注重实际编程能力的培养，适合作为普通高等院校应用型本科各相关专业的C++程序设计课程的教材，也适合作为编程开发人员的培训或自学用书。

图书在版编目(CIP)数据

C++面向对象程序设计——基于Visual C++2010/吴克力编著. --北京：清华大学出版社，2013
(高等学校应用型特色规划教材)
ISBN 978-7-302-31791-3

Ⅰ. ①C…　Ⅱ. ①吴…　Ⅲ. ①C语言—程序设计—高等学校—教材　Ⅳ. ①TP312

中国版本图书馆CIP数据核字(2013)第059865号

责任编辑：章忆文　杨作梅
封面设计：杨玉兰
责任校对：周剑云
责任印制：刘海龙

出版发行：清华大学出版社
网　　址：http://www.tup.com.cn，http://www.wqbook.com
地　　址：北京清华大学学研大厦A座　　邮　　编：100084
社 总 机：010-62770175　　邮　　购：010-62786544
投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn
质 量 反 馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
课 件 下 载：http://www.tup.com.cn，010-62791865

印 装 者：北京国马印刷厂
经　　销：全国新华书店
开　　本：185mm×260mm　　**印　　张**：25.5　　**字　　数**：616千字
版　　次：2013年4月第1版　　**印　　次**：2013年4月第1次印刷
印　　数：1～3000
定　　价：39.80元

产品编号：051845-01

丛 书 序

二十一世纪人类已迈入“知识经济”时代，科学技术正发生着深刻的变革，社会对德才兼备的高素质应用型人才的需求更加迫切。如何培养出符合时代要求的优秀人才，是全社会尤其是高等院校面临的一项急迫而现实的任务。

为了培养高素质应用型人才，必须建立高水平的教学计划和课程体系。在教育部有关精神的指导下，我们组织全国高校计算机专业的专家教授组成《高等学校应用型特色规划教材》学术编审委员会，全面研讨计算机和信息技术专业的应用型人才培养方案，并结合我国当前的实际情况，编审了这套《高等学校应用型特色规划教材》丛书。

编写目的

配合教育部提出要有相当一部分高校致力于培养应用型人才的要求，以及市场对应用型人才需求量的不断增加，本套丛书以“理论与能力并重，应用与应试兼顾”为原则，注重理论的严谨性、完整性，案例丰富、实用性强。我们努力建设一套全新的、有实用价值的应用型人才培养系列教材，并希望能够通过这套教材的出版和使用，促进应用型人才培养的发展，为我国建立新的人才培养模式做出贡献。

已出书目

本丛书陆续推出，滚动更新。现已出版如下书目：

- Visual Basic 程序设计与应用开发
- Visual FoxPro 程序设计与应用开发
- 中文 Visual FoxPro 应用系统开发教程(第二版)
- 中文 Visual FoxPro 应用系统开发上机实验指导(第二版)
- Delphi 程序设计与应用开发
- 局域网组建、管理与维护
- Access 2003 数据库教程
- 计算机组装与维护
- 多媒体技术及应用
- 计算机网络技术
- Java 程序设计与应用开发
- Visual C++程序设计与应用开发
- Visual C# .NET 程序设计与应用开发
- C 语言程序设计与应用开发
- 计算机网络技术与应用
- 微机原理与接口技术
- 微机与操作系统贯通教程
- Windows XP+Office 2003 实用教程
- C++程序设计与应用开发

- ASP.NET 程序设计与应用开发
- Windows Vista + Office 2007 + Internet 应用教程
- 计算机应用基础(等级考试版 • Windows XP 平台)
- Java 程序设计与应用开发(第 2 版)
- Internet 实用简明教程
- 大学生计算机科学基础(上、下册)

丛书特色

- 理论严谨，知识完整。本丛书内容翔实、系统性强，对基本理论进行了全面、准确的剖析，便于读者形成完备的知识体系。
- 入门快速，易教易学。突出“上手快、易教学”的特点，用任务来驱动，以教与学的实际需要取材谋篇。
- 学以致用，注重能力。将实际开发经验融入基本理论之中，力求使读者在掌握基本理论的同时，获得实际开发的基本思想方法，并得到一定程度的项目开发实训，以培养学生独立开发较为复杂的系统的能力。
- 示例丰富，实用性强。以实际案例和部分考试真题为示例，兼顾应用与应试。
- 深入浅出，螺旋上升。内容和示例的安排难点分散、前后连贯，并采用循序渐进的编写风格，层次清晰、步骤详细，便于学生理解和学习。
- 提供教案，保障教学。本丛书绝大部分教材提供电子教案，便于老师教学使用，并提供源代码下载，便于学生上机调试。

读者定位

本系列教材主要面向普通高等院校和高等职业技术院校，适合本科和高职高专教学需要；同时也非常适合编程开发人员培训、自学使用。

关于作者

丛书编委特聘请执教多年、有较高学术造诣和实践经验的名师参与各册的编写。他们长期从事有关的教学和开发研究工作，积累了丰富的经验，对相应课程有较深的体会及独到的见解，本丛书凝聚了他们多年的教学经验和心血。

互动交流

本丛书贯穿了清华大学出版社一贯严谨、科学的图书风格，但由于我国计算机应用技术教育正在蓬勃发展，要编写出满足新形势下教学需求的教材，还需要我们不断地努力实践。因此，我们非常欢迎全国更多的高校老师积极加入到《高等学校应用型特色规划教材》学术编审委员会中来，推荐并参与编写有特色、有创新的应用型教材。同时，我们真诚希望使用本丛书的教师、学生和读者朋友提出宝贵意见或建议，使之更臻成熟。联系信箱：Book21Press@126.com。

《高等学校应用型特色规划教材》学术编审委员会

《高等学校应用型特色规划教材》
学术编审委员会

前　言

C++程序设计语言从20世纪80年代推出，至今已有近30年的历史，是一种灵活、高效、应用面广、面向对象的计算机编程语言。时至今日，C++依然在系统软件、游戏、网络和嵌入式等领域中广泛应用，是主流的程序设计语言之一。

目前，高等学校的计算机及相关专业普遍选C++作为计算机编程的入门语言进行教学，此外，许多理工类专业也开设了该课程。C++是在结构化的C语言之上引入面向对象技术演变而来的。对于初学者，学习C++语言是否需要先学习C语言呢？事实上，许多C++程序设计教程也是先讲结构化的C语言部分，后讲面向对象的C++技术。在多年的教学实践中，作者发现对于初学者来说，结构化程序设计方法的学习会对面向对象设计技术的掌握产生负面影响。例如，在学习类的概念时，受结构化程序中函数调用需要传递实参的影响，许多学生不习惯直接访问类中已封装的数据，常常试图将类中的数据传递给成员函数。结构化程序设计思想和方法学习得越好，影响就越大。实践证明，在有限的教学时间内，直接学习面向对象的C++编程技术更有利于概念的掌握和技能的提升。面向对象是当今主流的编程技术，例如流行的Java和C#均是面向对象的程序设计语言。学好C++的面向对象编程技术，无疑能为学习Java、C#打下扎实的基础。

本书在编写过程中，先后参阅了多部国内外C++程序设计类书籍，从中吸收了许多新的思想、方法和知识，并结合作者多年的教学实践和软件开发经验，博采百家之长，力求有所创新，并形成特色。本书具有以下特点。

(1) 以面向对象技术为核心，循序渐进，强化编程技能的培养。本书在介绍数据类型、基本运算、程序的控制结构和函数等知识之后，即引入类的概念，并在其后的例程中强化用类设计程序，将封装的思想方法及早地传授给学习者。考虑到学习有一个由浅入深、逐步提高的过程，本书将较难的知识点尽可能早引入，并通过后继章节的反复应用，不断强化，达到能够灵活运用的目标。为避免因案例过于简单而不能很好地体现面向对象编程思想和技术的优势，书中给出了多个相对复杂的综合示例，以此演示C++面向对象程序设计的方法。

(2) 利用调试跟踪工具剖析关键知识点，化抽象为直观，强化基本概念的掌握。C++中的许多概念和技术比较抽象、难懂，学习难度大。用调试工具分析和讲授C++中的概念，是一种值得推荐的直观教学法。在教学中，借助这种教学方法能演示程序执行的机理，搞清语法规则的“之所以然”，具有事半功倍的效果。本书许多例程的后面撰写了“跟踪与观察”，其中包含程序在调试运行时跟踪窗口的截图，旨在通过直观的解析帮助读者理解并掌握一些重要的概念和语法规则。此外，尽快地学会调试工具的使用，还有助于初学者提高编程能力和掌握排除错误的能力。

(3) 设计基于C++/CLI的窗体应用程序，与时俱进，强化实际应用的能力。目前多数教材编写程序时仍基于曾经十分流行的Visual C++ 6.0开发平台，而微软公司的C++开发平台经过几次升级，已推出最新的Visual C++ 2010，早期的Visual C++ 6.0平台在实际应用开发中正逐渐淡出。在Visual C++ 6.0中开发Windows窗体应用程序时使用MFC类库，

虽然在 Visual C++ 2010 版本中依然支持用 MFC 开发窗体应用程序，但随着技术的进步，用多种语言设计运行于.NET 框架上的窗体应用程序已成为主流。为适应技术发展潮流，本书在重点介绍 C++/CLI 语言之后，通过若干个小应用程序示例学习窗体应用程序的设计方法。C++/CLI 语言中的许多新的概念是基于.NET 框架的，与 C#语言十分相似，体现了面向对象技术的发展。Visual C++ 2010 类似于 Delphi、VB 的快速应用程序设计(RAD)方法，能简化应用程序界面的设计，降低开发难度，提升初学者的学习兴趣。

(4) 内容全面，语言简练，示例丰富。书中内容涵盖了用 C++面向对象技术进行程序设计所需的基础知识和技能。在语言表述上，尽可能地简洁、准确、有条理，以便于阅读和理解。全书共有 130 多个示例程序，这些程序编写规范，可模仿性好。

本书共分 13 章，包括标准 C++和 C++/CLI 两大部分。第 1~10 章为标准 C++语言，主要内容有数据类型、基本运算、程序的控制结构、函数、类与对象、继承、多态、动态内存、模板、异常处理和流等基本概念及编程技术。第 11~12 章介绍 C++/CLI 和 WinForm 窗体应用程序的设计技术。第 13 章为项目实践。

在教学过程中，根据具体的教学课时数，下列章节可以不讲或者安排自学：5.6 节“函数指针”、8.4 节“标准模板库简介”、10.6 节“字符串流”、第 11 章“C++/CLI 程序设计基础”和第 12 章“WinForm 应用程序设计”。

由于作者水平有限，书中不足之处在所难免，敬请读者不吝批评指正。

编　者

目　录

第 1 章　C++语言概述

C++语言从诞生至今已有近 30 年的历史，是流行时间长、应用面广的一门计算机程序设计语言。本章主要介绍 C++语言的发展历史、面向对象技术的基础知识、运用 Visual C++ 2010 开发平台进行程序设计和调试程序的基本方法。

1.1　C++程序设计语言简介

C++程序设计语言在发展过程中与时俱进，成为开发各种系统和应用软件的主要语言。本节首先简要介绍 C++语言的发展历程以及面向对象技术的一些基本概念，最后结合作者多年的教学经验，列出学好 C++程序设计课程的几点建议。

1.1.1　C++语言的发展历程

C++程序设计语言是从 C 语言发展而来的，C 语言起源于美国 AT&T 贝尔实验室。1969 年 Ken Thompson 为 DEC PDP-7 计算机设计了一个操作系统软件，就是最早的 Unix。之后，Ken Thompson 又根据剑桥大学 Martin Richards 设计的 BCPL 语言为 Unix 设计了一种便于编写系统软件的语言，命名为 B 语言。B 语言是一种无类型的语言，直接对机器字操作，这一点与后来的 C 语言有很大不同。作为系统软件编程语言的第一个应用，Ken Thompson 使用 B 语言重写了其自身的解释程序。1972~1973 年，Ken Thompson 与同在贝尔实验室的 Denis Ritchie 改造了 B 语言，为其添加了数据类型的概念，并将原来的解释程序改写为可以直接生成机器码的编译程序，然后将其命名为 C 语言。1973 年，Ken Thompson 小组在 PDP-11 机上用 C 语言重新改写了 Unix 内核。与此同时，C 语言的编译程序也被移植到 IBM 360/370、Honeywell 11 以及 VAX-11/780 等多种计算机上，迅速成为应用最广泛的程序设计语言。

20 世纪 80 年代初，贝尔实验室的 Bjarne Stroustrup 博士及其同事开始针对 C 语言的类型检查机制相对较弱、缺少支持代码重用的语言结构等缺陷进行改进和扩充，形成了带类的 C(C with class)，即 C++最早的版本。后来，Stroustrup 和他的同事们又为 C++引进了运算符重载、引用、虚函数等许多特性，并使之更加精炼，于 1989 后推出了 AT&T C++ 2.0 版。随后美国国家标准化协会 ANSI 和国际标准化组织 ISO 一起进行了标准化工作，并于 1998 年正式发布了 C++语言的国际标准。

1998 标准发布后的几年里，委员会处理各种缺陷报告，并于 2003 年发布了一个 C++ 标准的修正版本。此后的标准更新原定是在 2009 年，但是由于对于新特性的争论激烈，完整的新标准至今仍没有形成。

C++语言在发展过程中不断地从其他计算机程序设计语言中吸收养分，其功能越来越强大，复杂性也在不断增加，已经成为当今主流程序设计语言中最复杂的一员。

1.1.2 面向对象程序设计技术

早期的软件开发是使用机器语言或者汇编语言在特定的机器上进行设计和编写，软件规模比较小，也不需要使用系统化的软件开发方法，基本上是个人设计编码、个人操作使用的模式。

从20世纪60年代中期开始，大容量、高速的计算机问世了，计算机应用面扩大，程序设计的复杂度也随之增长，出现了软件开发费用高和进度失控、软件的可靠性差、生产出来的软件难以维护等问题。1968年，计算机科学家在联邦德国召开国际会议，正式提出了“软件危机”的概念，以及“软件工程”一词，从此，一门新兴的工程学科——软件工程学应运而生。

1965年，E.W.Dijikstra提出了采用自顶向下、逐步求精的程序设计方法，指出程序设计可以使用三种基本控制结构来构造程序，任何程序都可由顺序、选择、循环三种基本控制结构进行构造。

结构化程序设计(Structured Programming)思想的提出是为解决当时程序设计中由于使用goto语句造成程序流程混乱、理解和调试程序困难的问题，它强调以模块功能和处理过程设计为程序设计原则。

结构化程序设计方法的基本思想是，把一个复杂问题的求解过程分阶段进行，每个阶段处理的问题都控制在人们容易理解和处理的范围内。支持结构化程序设计的高级语言在20世纪70年代初相继诞生，其典型代表有Pascal语言、C语言。

结构化程序设计是一种面向过程的程序设计方法，它把数据和处理数据的操作分离为相互独立的实体。当数据结构改变时，所有相关的处理过程都要进行相应的修改，每一种相对于旧问题的新方法都要带来额外的开销，程序的可重用性差。随着图形用户界面操作系统的出现，程序运行由顺序运行演变为事件驱动，软件使用变得越来越方便，但开发起来却越来越困难，软件的功能很难用结构化方法来描述和实现，使用面向过程的方法来开发和维护程序都变得非常困难。

面向对象程序设计(Object Oriented Programming)以对象作为程序的基本单元，将数据和操作封装其中，以提高软件的重用性、灵活性和扩展性。面向对象程序设计是以一种更接近于人类认知事物的方法建立模型，以对象作为计算主体，对象拥有自己的名称、状态以及接受外界消息的接口。在对象模型中，产生新对象、销毁旧对象、发送消息、响应消息就构成OOP计算模型的根本。面向对象程序设计比结构化程序设计更具有创建可重用代码和更好地模拟现实世界环境的能力。

在面向对象程序设计中，对象是要研究的任何事物。现实世界的诸多有形的实体(如书、汽车、人、商店、图形等)都可看作对象。此外，一些抽象的规则、计划或事件也能表示为对象。对象由数据(描述事物的属性)和作用于数据的操作(体现事物的行为)构成一独立的整体。从程序设计者来看，对象是一个程序模块；从用户来看，对象为他们提供所希望的行为。

类是对一组有相同数据和操作的对象的抽象，一个类所包含的方法和数据描述了一组对象的共同属性和行为。对象则是类的具体化，是类的实例。在面向对象程序设计中，经常用已有的类派生新类，并形成类的层次结构。

消息是对象之间进行通信的一种规格说明。一般它由 3 个部分组成：接收消息的对象、消息名及实际变元。

面向对象程序的设计方法具有如下 3 个特点。

(1) 封装性。封装是一种信息隐蔽技术，通过类的说明实现封装，是对象的重要特性。封装使数据和加工该数据的方法(函数)组合为一个整体，以实现独立性很强的模块，使得用户只能见到对象的外特性(对象能接受哪些消息，具有哪些处理能力)，而对象的内特性(保存内部状态的私有数据和实现加工能力的算法)对用户是隐蔽的。封装的目的在于把对象的设计者和对象的使用者分开，使用者不必知晓行为实现的细节，只需用设计者提供的消息来访问该对象。

(2) 继承性。继承性体现在类的层次关系中，派生的子类拥有父类中定义的数据和方法。子类直接继承父类的全部描述，同时可修改和扩充，并且继承具有传递性。继承分为单继承(一个子类仅有一父类)和多重继承(一个子类可有多个父类)。继承不仅为软件系统的设计带来代码可重用的优势，而且还增强了系统的可扩充性。

(3) 多态性。对象根据所接收的消息而做出动作。同一消息为不同的对象接受时可产生完全不同的行动，这种现象称为多态性。利用多态性，用户可发送一个通用的信息，而将所有的实现细节都留给接受消息的对象自行决定，这样，同一消息即可调用不同的方法。

多态性的实现受到继承性的支持，利用类继承的层次关系，把具有通用功能的声明存放在类层次中尽可能高的地方，而将实现这一功能的不同方法置于较低层次，这样在这些低层次上生成的对象就能给通用消息以不同的响应。C++语言通过在派生类中重定义基类函数(定义为重载函数或虚函数)来实现多态性。

1.1.3　学习 C++程序设计的注意事项

初学者学习 C++程序设计语言普遍感到困难，特别是在大学一年级，由于新生学习能力不强，还不能适应大学学习，使得该课程的教与学难度均较大。C++程序设计是一门强调动手实践的课程，缺乏一定量的编程练习是导致学习效果不佳的原因之一。初学者进行编程首先遇到的问题是：一个简单的程序在运行时可能会出现一大堆错误信息，而自己又不知怎样解决。一个程序练习题在机器上调试两个小时甚至更长时间也没能完成，从而失去信心。“能听懂，但我不会做。”这是许多初学者的反馈，究其原因，还是编程训练不到位。

培养程序设计能力如同学习写作文，需要相当一段时间的学习、积累和磨练才能达到一定的水平。在本科教学阶段，C++程序设计课程的后继课程(如数据结构、操作系统、编译原理等)将继续对编程能力进行训练。程序员的编程水平都要经历从初级至中级，再到高级这样一个发展的过程。

相对于其他程序设计语言，C++语言的概念和技术不仅多，并且比较难以掌握。怎样才能用较短的时间学好 C++语言，并具有一定的程序设计能力呢？下面的几个注意点对学习者会有帮助，供学习时参考。

(1) 深刻理解语法规则的内涵。学习程序设计语言少不了学语法规则，学习者应不为语法的表象所迷惑，而应追求理解语法规则后面隐藏的东西，即计算机在运行程序时所做的操作。

(2) 多读优秀的程序段，从中学习程序设计技巧。积累的编程技巧越多，程序设计能力的提高也就越快。这里应注意不要死记硬背，要活学活用。

(3) 少做纸上的程序填空类题目，多在计算机上做编程练习题。初期以输入并调通完整的例程为主，达到熟悉编程环境和练习调试方法的目的，期间可穿插完成一些简单的程序练习题。熟悉工具之后，尽可能在计算机上独立完成书中的编程练习题，坚持每天都编程，记住熟能生巧！编程能力是通过不断练习磨练出来的，正如人的肌肉需要通过一定量的体育锻炼才能发达一样。

(4) 重点学习面向对象技术，学会用面向对象思想分析和描述问题。C++语言是 C 语言的超集，其中包含了结构化和面向对象程序设计两种方法。在教学中发现，对于初学者，如果先学结构化程序设计，再学面向对象程序设计方法，由于先入为主的因素，反而会对掌握面向对象技术产生负面的影响。

(5) 注意逐渐养成良好的程序设计风格。程序的可读性、健壮性、易扩展性和易维护性非常重要，不要因追求所谓的技巧而编写难懂的程序。可读性是第一位的。要深刻地认识到：写程序不仅仅是让计算机完成某一项任务的，而且还是让人来阅读的。

(6) 根据自己的兴趣爱好，完成一个小型的应用软件项目。边学边做，边做边学。可以这样说：软件完成之日，也是你 C++语言学成之时。

1.2 Visual C++ 2010 编程工具简介

微软公司早在 1998 年推出的 Visual C++ 6.0(VC 6.0)是一款流行面广、业界使用时间长的软件开发工具，目前还有许多教材选用它作为 C++语言教学的软件平台。

随着新标准的推出和软件技术的发展，VC 6.0 对新标准和新操作系统的支持问题愈发明显，微软公司后来推出了多个 Visual C++版本，目前最新的版本是 Visual C++ 10.0，即 Visual C++ 2010。本书选用 Visual C++ 2010 为教学软件平台，旨在让初学者能够直接接触新的技术和工具，不在过时的软件平台上花费时间。

1.2.1 C++程序生成过程

C++语言是一种面向对象的高级程序设计语言，高级语言编写的程序是不能直接被计算机识别的，必须经过转换才能被执行。高级语言转换为计算机可识别的机器语言的方式主要有两种：一种是解释执行方式，它类似于英语翻译成汉语时采用的同声翻译，应用程序源代码一边由相应语言的解释器翻译成目标代码(机器语言)，一边执行，因此效率比较低，而且不能生成可独立执行的可执行文件，应用程序不能脱离其解释器，但这种方式比较灵活，可以动态地调整、修改应用程序；另一种是编译方式，编译是指在应用源程序执行之前，就将程序源代码翻译成目标代码，它类似于英语翻译成汉语时所采用的笔译方法，因此其目标程序可以脱离其语言环境独立执行，使用比较方便、效率较高，但修改应用程序很不方便。Visual C++在生成非托管代码(参见 1.2.2 小节)时采用编译方式，在生成托管代码时采用的是先编译成中间语言代码，再由.NET 框架的公共语言运行时解释执行。

下面简要介绍用 C++语言设计应用程序所经历的几个阶段。

(1) 编写程序。在文本编辑器中用 C++语言编写源代码文件，以扩展名.cpp 保存源代码程序。

(2) 程序预处理。源程序在被编译之前，先由预处理器根据源代码中的预处理指令在源代码中进行相应的插入与替换字符文本的操作。

(3) 编译程序。编译器将 C++程序翻译成目标代码(本地代码)。如果是在.NET 平台上运行的程序，编译器则将程序编译成中间语言代码(托管代码)。

(4) 连接程序。程序通常包含对标准或其他类库所定义的函数和数据的调用，连接器将被调用的相关代码组合到可执行文件中。最后生成的可执行文件的扩展名为.exe，这是一个在操作系统中或.NET 框架上可运行的程序。

(5) 运行程序。由操作系统加载可执行文件，将其先读入到计算机内存中，最后 CPU 根据程序中的指令完成各种操作。

(6) 调试程序。程序在编译、连接和运行阶段都可能出现错误，程序员需要用系统提供的调试工具帮助发现并指出错误及原因，修改源程序中的错误。

1.2.2 .NET 框架与 Visual C++ 2010

2000 年，微软执行总裁比尔·盖茨提出了.NET 战略，这是微软面向未来互联网的战略，它包含了一系列关键技术，使程序员可以更快、更方便地开发应用程序。

从技术角度看，.NET 应用是运行于.NET 框架之上的应用程序。.NET 框架是微软.NET 技术的核心，历经数年的发展，.NET 框架从 1.0 版到目前最新的 4.0 版，使得.NET 已经发展成为构建企业应用程序最重要的平台之一。.NET 框架提供了托管执行环境、简化的开发和部署以及与各种编程语言的集成，是支持生成和运行下一代应用程序和 Web 服务的内部 Windows 组件。.NET 框架的关键组件为公共语言运行时(Common Language Runtime)和.NET 框架类库，该类库包括 ADO.NET、ASP.NET、Windows 窗体和 Windows Presentation Foundation。

公共语言运行时(CLR)与 Java 的虚拟机一样，是一个运行时环境，它负责计算机内存的分配和回收等资源管理工作，并保证应用软件和底层操作系统之间必要的分离。在 CLR 上运行的程序通常称为“托管的”(Managed)代码，不在 CLR 上而是直接在计算机 CPU 上运行的程序被称为“非托管的”(Unmanaged)代码(又称本地代码)。在 CLR 上运行的程序先由编译器生成不能在 CPU 上直接运行的中间语言(Intermediate Language)代码，在代码被调用执行时，由 CLR 装载应用程序的中间语言代码至内存，再通过即时(Just-In-Time)编译技术将其编译成能在所运行的计算机上直接被 CPU 执行的本地代码。这种技术在 20 世纪 80 年代早期的商业软件 Smalltalk 上已实现，目前 Java 虚拟机的实现中使用了这一技术。

在 Visual C++ 2010 编程环境中，既可以用标准 C++语言编写在 CPU 上直接运行的被编译为本地代码的应用程序，也支持编写能在 CLR 中运行的被编译成中间语言代码的程序。微软公司还专门设计了一门与标准 C++兼容的计算机语言 C++/CLI(Common Language Infrastructure)，用于支撑在.NET 框架上采用 C++语言开发应用软件。Visual C++ 2010 并不强迫程序员编写的程序是用托管代码还是非托管代码，而且允许程序员在同一个项目中不同程序之间，甚至在同一个文件内混合使用托管代码和非托管代码。

本书在讲授标准 C++语言部分所编写的例程时，全部采用控制台应用程序方式实现，