

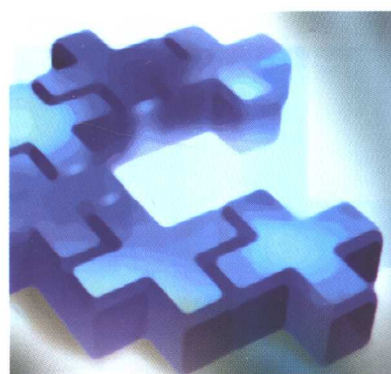
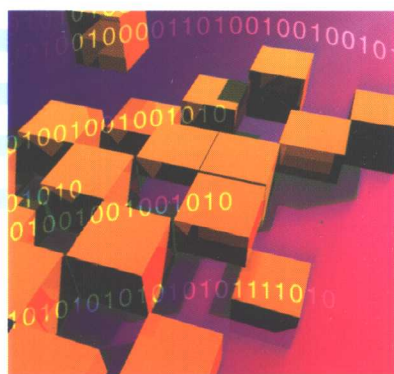
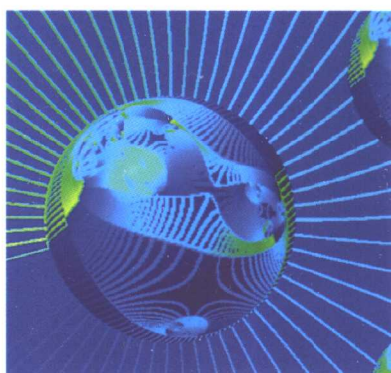


21 世纪高等学校应用型教材

C++ 程序设计

□ 余苏宁 主 编

□ 王明福 副主编



高等教育出版社
Higher Education Press

内容提要

全书系统地介绍了 C/C++ 语言的基础知识、基本语法以及编程方法,深入浅出地讲述了 C++ 语言面向对象的重要特征(抽象、封装、继承等)。全书共 13 章,主要内容包括:概论;数据类型、运算符与表达式;程序的结构与语句;数组与字符串;函数;指针;构造数据类型;非面向对象的拓展;类;继承与派生;多态与动态联编;运算符重载的实例;模板。

本书的特点是定位准确、选材适当,实例丰富、重点突出,特别适合作为应用型本科、高职高专院校计算机及相关专业学生学习 C++ 程序设计的教材,也可作为其他理工科专业 C++ 程序设计的教材,也适合作为相关技术人员的自学参考书。

书中所有程序全部运行通过,所有程序源代码及相关文档,包括综合性示例程序(时钟、人工生命、带回朔的迷宫探索)在两种开发环境(Borland C++3.1 和 Visual C++6.0)下的创建过程说明和源程序清单以及有关的教学资料(包括教学电子教案),均可以从高等教育出版社网站上下载,网址为:[http:// cs.hep.com.cn](http://cs.hep.com.cn) 或 <http://www.hep.edu.cn>。

图书在版编目(CIP)数据

C++ 程序设计/余苏宁主编. — 北京:高等教育出版社,2004.1

ISBN 7-04-012303-7

I. C... II. 余... III. C 语言—程序设计
IV. TP312

中国版本图书馆 CIP 数据核字(2003)第 121650 号

出版发行 高等教育出版社
社 址 北京市西城区德外大街 4 号
邮政编码 100011
总 机 010-82028899

购书热线 010-64054588
免费咨询 800-810-0598
网 址 <http://www.hep.edu.cn>
<http://www.hep.com.cn>

经 销 新华书店北京发行所
印 刷 北京民族印刷厂

开 本 787×1092 1/16
印 张 18.5
字 数 450 000

版 次 2004 年 1 月第 1 版
印 次 2004 年 1 月第 1 次印刷
定 价 24.00 元

本书如有缺页、倒页、脱页等质量问题,请到所购图书销售部门联系调换。

版权所有 侵权必究

前 言

由于计算机技术的迅猛发展,极大地促进了计算机应用的普及和深入,致使软件系统的规模和复杂性急剧膨胀,传统的结构化程序设计已不能满足社会发展的需要。巨大的需求引发了软件技术的革命,诞生了面向对象程序设计的思想和方法。作为这种新技术的典型代表,C++语言以其完全支持面向对象程序设计、与C语言高度兼容、高效灵活、功能强大、可重用性好等优点得到业内外广泛重视、推崇和应用。

本书以面向对象的基本思想和方法为主要内容,将重点放在面向对象基本特征和基本技术的学习上。全书可分为两大部分。第1章~第7章是基础部分,主要介绍C++程序设计语言的基本要素、程序的结构和过程化基础。第8章~第13章是C++对C的扩展和面向对象程序设计部分,主要讲述了C++面向对象程序设计中的类、对象、继承、派生、多态、虚拟函数和动态联编等核心内容。

全书以“实例引入,改造拓宽,项目综合”的方式组织内容,给出了大量的实例,所有的实例都给出了运行结果,看到结果就如同看到程序运行一样。所有的理论知识都以实例的演示、分析、说明、归纳总结来讲解,使抽象的理论变得更具体、更实际。

在使用本书时,应注意下面的问题:

对于学过C语言的读者来说,第一部分的内容主要起复习强化作用。其中,第1章~第4章可快速浏览;第5章~第7章由于涉及C/C++语言的难点,应再细致深入地复习一遍,为后面的学习打下良好的基础。

对于C++教学平台的选择,条件好的可采用Visual C++ 6.0。但要强调的是,只需要利用Visual C++的基本功能,对于其他功能不做过多要求,书中涉及的绘图功能也只限于几条语句的使用,深入学习Visual C++是后续课程的任务。如果条件有限,也可选用Borland C++ 3.1。它是面向DOS和Windows的系统,使用较为方便。

本书凝聚了作者多年C/C++程序设计教学及软件应用系统开发的经验,选材适当,定位准确,实例丰富,深入浅出,重点突出,具有很强的实用性和可读性。

本书适合作为应用型本科、高职高专院校计算机及相关专业学生学习C++程序设计的教材,也可作为其他理工科专业C++程序设计的教材,也适合作为相关技术人员的自学参考书。

书中所有程序全部运行通过,所有程序源代码及相关文档,包括综合性示例程序(时钟、人工生命、带回溯的迷宫探索等)在两种开发环境(Borland C++ 3.1和Visual C++ 6.0)下的创建过程说明和源程序清单以及有关的教学资料(包括教学电子教案),均可以从高等教育出版社网站上下载,网址为:<http://cs.hep.com.cn>或<http://www.hep.edu.cn>。

本书由余苏宁主编,王明福任副主编。其中第2章由王明福编写,其余各章由余苏宁编写,最后由余苏宁负责全书的统稿、定稿。此外,孙甄、余轶、孙贤、肖红梅在文稿的录入、整理方面做了大量工作;齐治文、聂哲、李亮、乌云高娃、杨淑萍、刘文新对本书的程序调试和查错做了大量的工作;李俊平和王梅等对本书的编写思路、实例的选取和运用提出了许多宝贵意见和建议,并承

担了部分章节的文字录入与审核工作;在此一并表示衷心的感谢。

由于编者水平有限,加之时间仓促,书中难免有错漏之处,敬请广大读者批评赐教,在此先表谢意。编者的 E-mail 地址是:yusuning@oa.szpt.net 或 yusuning@21cn.com。

编 者

2003 年 12 月于深圳

目 录

第1章 概论	(1)	2.3.9 算术、关系、逻辑、赋值混合运算 ...	(43)
1.1 C++ 程序演示	(1)	2.3.10 运算顺序与副作用	(44)
1.2 C++ 程序的结构	(3)	习题二	(45)
1.2.1 简单的 C++ 程序	(3)	第3章 程序的结构与语句	(46)
1.2.2 C++ 程序的结构	(4)	3.1 程序结构与语句概述	(46)
1.2.3 C++ 源程序的编辑、编译、链接 和运行	(6)	3.2 顺序结构	(47)
1.3 C++ 集成开发环境的安装与运行	(7)	3.3 选择结构	(50)
1.3.1 系统配置	(7)	3.3.1 if 语句(条件语句)	(50)
1.3.2 Visual C++ 6.0 集成开发 环境的安装	(7)	3.3.2 switch 语句(开关语句)和 break 语句	(52)
1.3.3 Visual C++ 6.0 集成开发 环境的使用	(12)	3.4 循环结构	(54)
1.4 面向对象技术的形成、发展和 C++ 的诞生	(19)	3.4.1 当型循环(while 循环)	(55)
1.5 C++ 的主要特点概述	(20)	3.4.2 直到型循环(do-while 循环)	(55)
习题一	(21)	3.4.3 计数循环(for 循环)	(56)
第2章 数据类型、运算符与表达式	(22)	3.4.4 循环的嵌套	(57)
2.1 基本字符、标识符和关键字	(22)	3.4.5 continue 语句	(58)
2.1.1 基本字符	(22)	3.4.6 goto 语句和标号语句	(59)
2.1.2 标识符	(23)	习题三	(59)
2.1.3 关键字(保留字)	(23)	第4章 数组与字符串	(61)
2.2 数据与数据类型	(24)	4.1 数组概述	(61)
2.2.1 常量和变量	(24)	4.2 一维数组	(62)
2.2.2 基本类型	(26)	4.2.1 一维数组的定义和引用	(62)
2.2.3 数据类型转换	(33)	4.2.2 一维数组的初始化	(63)
2.3 表达式与运算符	(35)	4.2.3 一维数组应用举例	(63)
2.3.1 赋值运算符	(36)	4.3 二维数组和多维数组	(64)
2.3.2 算术运算符	(37)	4.3.1 二维数组的定义和引用	(64)
2.3.3 自增(减)运算符	(38)	4.3.2 二维数组的初始化	(65)
2.3.4 关系运算符	(39)	4.3.3 二维数组应用举例	(67)
2.3.5 逻辑运算符	(40)	4.4 字符型数组与字符串	(69)
2.3.6 条件运算符	(41)	4.4.1 字符数组的定义	(70)
2.3.7 逗号运算符	(41)	4.4.2 对字符数组初始化	(70)
2.3.8 位运算符	(42)	4.4.3 字符串	(70)
		4.4.4 字符型数组应用举例	(72)
		习题四	(74)
		第5章 函数与预处理	(75)

5.1 函数定义和声明	(75)	7.1 结构型	(138)
5.1.1 函数的定义	(76)	7.1.1 结构型、结构型变量及结构型 指针的定义	(139)
5.1.2 函数的声明	(77)	7.1.2 结构型的嵌套和匿名结构型	(140)
5.2 参数种类和参数传递方式	(78)	7.1.3 访问结构型变量	(141)
5.3 函数调用	(78)	7.1.4 结构型变量的初始化	(142)
5.4 函数的嵌套调用和递归调用	(80)	7.1.5 结构型数组	(142)
5.4.1 函数的嵌套调用	(80)	7.1.6 结构、指针与函数	(145)
5.4.2 函数的递归调用和递归函数	(81)	7.1.7 综合应用	(147)
5.5 存储属性	(83)	7.2 共用型	(151)
5.5.1 作用域	(84)	7.2.1 共用型的定义和共用型变量、 指针的定义	(151)
5.5.2 存储类别	(85)	7.2.2 共用型变量、指针的使用	(151)
5.6 内部函数与外部函数	(90)	7.3 枚举型	(153)
5.6.1 内部函数	(90)	7.4 类型定义 (typedef)	(157)
5.6.2 外部函数	(90)	习题七	(158)
5.7 预处理	(91)	第8章 非面向对象的拓展	(160)
5.7.1 宏定义	(91)	8.1 单行注释	(160)
5.7.2 文件包含	(95)	8.2 块(分程序)内任何位置定义和声明	(160)
5.7.3 条件编译	(97)	8.3 强制类型转换	(161)
习题五	(99)	8.4 流式 I/O	(162)
第6章 指针	(100)	8.5 函数的缺省参数	(164)
6.1 指针和指针变量	(100)	8.6 内联函数	(166)
6.2 指针变量的定义和使用	(103)	8.7 函数重载	(167)
6.2.1 指针变量的定义和使用	(103)	8.8 引用	(168)
6.2.2 指针变量作函数参数	(105)	8.8.1 独立引用	(168)
6.2.3 指向指针的指针和指针所指向 的类型	(106)	8.8.2 引用参数	(170)
6.3 指针与数组	(109)	8.8.3 返回引用的函数	(172)
6.3.1 数组的指针和数组元素的指针	(109)	8.9 const	(173)
6.3.2 指针与一维数组	(109)	8.9.1 const 类型说明符	(173)
6.3.3 指向数组的指针作函数参数	(113)	8.9.2 指针与常量	(174)
6.3.4 指针与二维数组	(114)	8.10 单目作用域运算符	(175)
6.3.5 动态数组与动态存储分配	(117)	8.11 动态存储分配与释放	(176)
6.3.6 指针数组	(120)	习题八	(180)
6.4 指针与字符串	(121)	第9章 类	(183)
6.5 指针与函数	(126)	9.1 类概念的引入和类的定义	(183)
6.5.1 返回指针值的函数	(126)	9.2 类作用域	(188)
6.5.2 指向函数的指针	(129)	9.3 数据成员与方法	(189)
6.5.3 指向函数的指针作函数参数	(130)	9.3.1 方法的定义和引用	(189)
6.5.4 main 函数的参数	(131)	9.3.2 内联方法	(190)
6.6 有关指针的数据类型小结	(133)	9.3.3 方法重载	(191)
习题六	(136)	9.3.4 const 对象和 const 成员函数	(195)
第7章 构造数据类型	(138)		

9.4.2 析构函数	(200)	11.4.2 产品管理程序的多态解	(251)
9.4.3 缺省构造函数	(201)	11.4.3 非多态解和多态解系统 的维护	(254)
9.4.4 拷贝构造函数	(202)	11.5 虚拟析构函数	(256)
9.5 成员对象和对象数组	(204)	11.6 动态联编的实现机理	(256)
9.5.1 成员对象	(204)	习题十一	(257)
9.5.2 对象数组	(206)	第 12 章 运算符重载	(258)
9.6 类与指针	(207)	12.1 引入运算符重载的必要性	(258)
9.6.1 指向类对象的指针	(207)	12.2 运算符重载的规则	(259)
9.6.2 this 指针	(208)	12.3 含有指针数据成员的运算符重载	(261)
9.6.3 指向类成员的指针	(210)	12.4 用户定义的转换	(262)
9.7 静态类成员	(213)	12.5 下标运算符重载	(265)
9.8 接口和实现的分离	(215)	12.6 运算符重载的实例	(266)
习题九	(219)	习题十二	(269)
第 10 章 继承与派生	(223)	第 13 章 模板	(270)
10.1 派生	(223)	13.1 类模板	(270)
10.1.1 基类、派生类与继承	(224)	13.2 函数模板	(272)
10.1.2 派生类的继承特性	(225)	13.3 特殊版本的模板	(273)
10.2 多重继承	(228)	13.4 模板友元	(274)
10.3 虚拟基类	(231)	13.5 编程示例	(276)
10.4 派生类构造函数的定义方法	(233)	习题十三	(278)
习题十	(235)	附录 A C/C++ 与 Borland C++ 3.1 中的 关键字表	(279)
第 11 章 多态与动态联编	(239)	附录 B 运算符的优先级和结合性	(281)
11.1 虚拟函数、动态多态性与动态联编	(239)	附录 C Borland C++ 3.1 独立图形运行 程序的建立	(283)
11.2 虚拟函数的访问属性	(242)	参考文献	(284)
11.3 纯虚拟函数与抽象类	(243)		
11.4 虚拟函数应用举例	(247)		
11.4.1 产品管理程序的非多态解	(247)		

第 1 章

概 论

本章导读

C++ 是在面向对象(OO, Object - Oriented)的大潮流中诞生的宠儿,同时由于它的广泛运用又极大地推动了面向对象技术的发展。C++ 是在 C 语言的基础上拓展而来的,引入了面向对象的机制,并且与 C 完全兼容,具有 C 高效灵活、功能强大、可移植性好等诸多优点,因此,一出世就受到广大软件工作者和软件开发商的青睐,版本不断翻新,功能不断完善,迅速成为面向对象程序设计(OOP, OO Programming)的最佳语言之一。

学习 C++, 至少可在下述两个方面受益:一是掌握一门新的编程语言,增长程序设计技能,为学习类似的其他语言(如 Java 语言、Visual C++、Visual J++、C# 等)打下良好的基础;二是以 C++ 为手段,通过编程实践可以更具体、更实际理解和掌握面向对象的一些基本特征(如:抽象、封装、继承、多态和动态联编等),为更好地学习和掌握面向对象程序设计的基本思想和技术打下良好的基础。

本章首先介绍一些有关 C++ 程序的基本知识、集成开发系统的安装和使用,然后简略地回顾面向对象技术的形成、发展和 C++ 的诞生过程,最后概括性地介绍 C++ 的主要特点。通过本章的学习,要求达到如下教学目标:

- 理解和掌握 C++ 程序的基本知识
- 掌握集成开发系统的安装和使用方法
- 了解面向对象技术的形成、发展和 C++ 的诞生
- 了解 C++ 的主要特点

1.1 C++ 程序演示

演示程序 1:时钟程序

程序运行时先以字符方式动态显示系统当前日期和时间,时间一秒一秒地变动,按任意一键后转换为以图形方式显示时钟,秒针一秒一秒地转动。再按任意一键后,又转换为以字符方式显示日期和时间,即按任意键为显示方式的交替,直到按 q 或 Q 键则结束程序的运行,如图 1.1 所示。

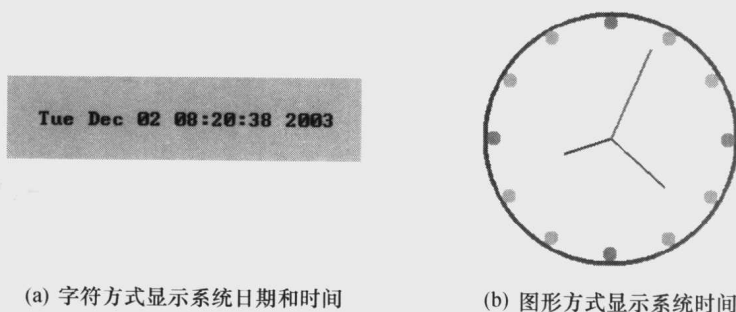


图 1.1 正在运行的时钟程序

演示程序 2: 音乐程序

程序演示时机内扬声器播放出“生日快乐”和“莫斯科郊外的晚上”乐曲。

演示程序 3: 人工生命

程序演示时在屏幕上显示出活动的生物体和食物(见图 1.2),生物体通过运动能吃到周围的食物而得以生存,当生存到一定的时间后,生物体可以繁殖,当生物体繁殖到一定数量时,由于食物有限,有些生物体得不到足够的食物而被饿死,或生命周期已到而死亡,生物体的数量开始减少,食物又丰富起来,生物体又开始增多……。

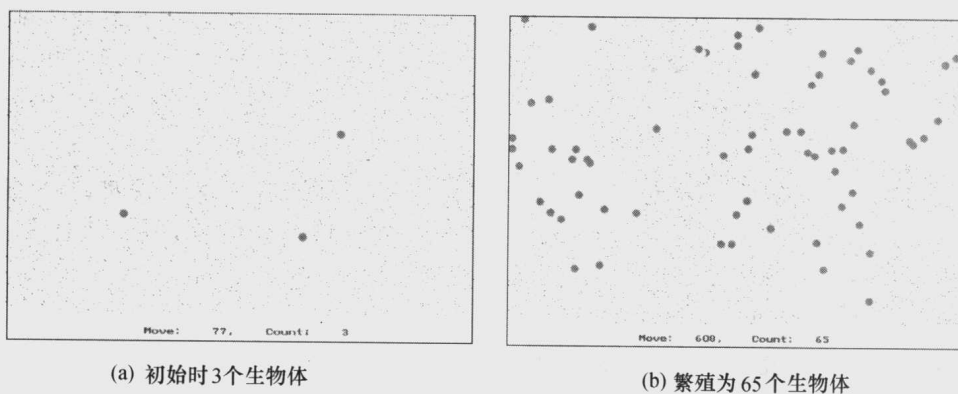


图 1.2 人工生命

以上程序是用 C++ 语言编写的,初步可见 C++ 语言的强大功能(从 <http://cs.hep.com.cn> 网站上可以下载演示程序 1 和程序 3 的 Visual C++ 版的程序编制过程和程序的主要代码清单)。通过本课程的学习相信大家都能编写出这样的程序。

演示程序 4: 迷宫程序

程序演示时在屏幕上显示一个复杂的迷宫以及探索者动态的探索过程(见图 1.3)。迷宫的大小、探索的策略和速度可以设定,迷宫的产生是随机的,因此每次生成的迷宫都不会重复,程序还可以设定是否显示回溯路线,以便更详细地了解回溯过程。

这个程序是用 Visual C++ 6.0 编写的,Visual C++ 是在 C++ 的基础上发展起来的面向对象的、具有程序自动生成功能的、可视化程序开发系统,它的基本语法仍然是 C++ 的,要想学好

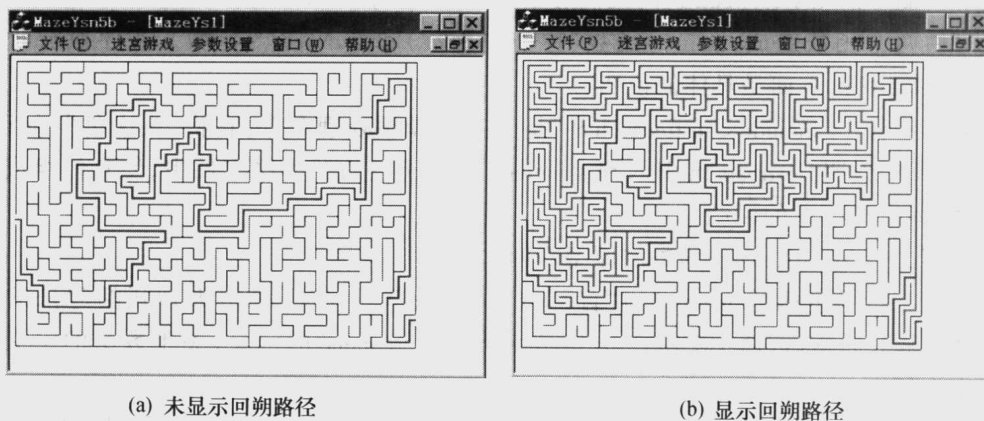


图 1.3 迷宫探索

Visual C++ 就必须先学好 C++。

1.2 C++ 程序的结构

1.2.1 简单的 C++ 程序

例 1.1 简单的 C++ 程序。

<pre>#include<iostream.h> void getName(char *p); void main() { char name[20]; getName(name); cout << " \nHello, " << name << "!\n "; } void getName(char *p) { cout << "What's your name ? "; cin >> p; }</pre>	<pre>//流式 I/O 必须包含 iostream.h 文件 //函数原型声明 //函数的首部 //函数体开始 //定义数组 //提示信息,接收输入 //向屏幕输出信息 //函数体结束 //由键盘输入</pre>
---	--

本程序运行时会在屏幕上显示如下信息：

What's your name ?

并等待您的输入,当您在键盘上输入如下信息：

ZhangSan

并按回车键后,屏幕上会显示出如下信息：

Hello, ZhangSan !

这是一个简单的 C++ 程序,它向我们展示了 C++ 程序的基本内容。

本程序由两个函数组成:main()函数(称为主函数)和 getName()函数。程序执行时首先从 main()函数开始运行,下面是具体的说明:

程序的第一行“#include <iostream.h>”是一条编译预处理命令,它将“头文件”(何谓“头文件”将在第2章中说明)iostream.h 引入到 #include 命令所在的源程序文件中,使它们组成一个文件以便编译(编译见下面的说明)。

程序的第二行是 getName()函数的原型声明。

程序的第三行是主函数 main()的“首部”(或称“头部”),接下来由大括弧括起来的部分是 main()函数的函数体,函数体中第一条语句是字符型数组 name 的定义(关于数组的定义和使用在第4章中介绍)。

函数体中第二条语句是调用 getName()函数的语句,当程序执行到此句时,控制转到 getName()函数中去执行。

该函数的第一条语句:

```
cout << "What's your name ? " ;
```

是向屏幕输出提示信息“What's your name ?”

第二条语句是接收键盘输入:

```
cin >> p;
```

键盘输入的信息(假定你输入的是“ZhangSan”)被存放在由字符指针 p 所指向的存储区中,即主函数 main()中定义的字符数组 name 中。

上两句中使用的“cout”和“cin”是“流”对象,分别代表屏幕和键盘。“<<”和“>>”是重载的运算符,分别用于输出和输入。关于“流”和“重载”后面的章节有专门的介绍。

getName()函数执行完后返回到主函数 main()中,继续执行 main()函数中调用 getName()函数的下一条语句:

```
cout << " \nHello, " << name << " ! \n " ;
```

该语句在屏幕新的一行上输出信息“Hello,”,然后再输出变量 name 的内容,即用户从键盘上输入的内容(“ZhangSan”),最后输出一个感叹号“!”并将光标置于下一行,于是在屏幕上显示出:

```
Hello, ZhangSan !
```

最后在主函数中结束整个程序的运行。

以上只是简单地介绍了这个程序的工作情况,更深入的知识将在后面逐步学习。

1.2.2 C++ 程序的结构

1. 从功能实现上看 C++ 程序的结构

从功能实现上看,C++ 程序的结构可归结如下:

(1) 函数是 C++ 程序的基本组成单位(确切地讲,应该是类、对象和对象的成员函数,除 main()函数外),每一个 C++ 程序都必须有且只能有一个 main()函数。main()函数是整个程序的入口,并完成对整个程序的控制作用,程序最后结束也在 main()函数中。这一点是和 C 语言程序

一样的。

除了 `main()` 函数外, C++ 程序中还可包含若干个其他函数。`main()` 函数调用其他函数, 其他函数之间也可以相互调用, 但其他函数不能调用 `main()` 函数。在例 1.1 中, 除了 `main()` 函数外, 还有一个 `getName()` 函数, `main()` 函数调用 `getName()` 函数完成在屏幕上显示提示信息并接收键盘输入的工作, 然后 `main()` 函数中输出信息并结束整个程序的运行。这些也是和 C 语言程序一样的。这是 C++ 面向过程的特点。

除此之外, C++ 与 C 的最大区别是: C++ 具有面向对象的特点。它将数据和函数封装成“类”, 用“类”定义“对象”, “对象”是面向对象程序的基本单位。关于面向对象的特征将在后面的章节中专门论述, 在此之前, 我们首先研究它的面向过程的特征。

(2) 一个函数由“函数的首部”和“函数体”两部分组成:

函数的首部即函数的第一行, 包括函数类型、函数名、函数参数(形参)名、参数类型等。例如:

void	getName	(char *	P)
↑	↑	↑	↑
函数类型	函数名	参数类型	参数名

一个函数名后面必须跟一对圆括弧, 函数参数可以没有, 如 `main()`。

函数体即函数首部下面的大括弧 `{……}` 的部分。如果一个函数内有多个大括弧, 则最外层的一对 `{ }` 为函数体的范围。

C 语言的函数体一般包括(C++ 中已没有声明部分和执行部分的区别了):

- 声明部分 在这部分中定义所用到的变量, 在 C 语言中声明部分必须在执行部分的前面, 但 C++ 中已经突破了这个规定。如例 1.1 中 `main()` 函数中的 `char name[20]`。在声明部分中还要对所调用的函数进行声明, 也可以在 `main()` 函数之外对所调用的函数进行声明, 如例 1.1 所示的 `void getName(char * p);`

- 执行部分 由若干个语句组成。

当然, 在某些情况下也可以没有声明部分(如例 1.1 中的 `getName()` 函数)。甚至可以既无声明部分, 也无执行部分, 但此时函数体的一对 `{ }` 仍然不能少, 如:

```
dump()
{ }
```

它是一个空函数, 什么也不干, 但这是合法的。

(3) C++ 程序中的函数是平行定义的, 不能嵌套定义, 它们被执行的先后次序不决定于它们在文件中的位置, 只决定于调用语句被执行的先后次序。`main()` 函数是由操作系统调用的, 不论 `main()` 函数放在整个程序中的什么位置(最前头, 或最后, 或在一些函数之前, 在另一些函数之后), 它总是首先被执行。

(4) C++ 程序书写格式自由, 一行内可以写几个语句, 一个语句可以分写在多行上。C++ 程序没有行号, 也不像 FORTRAN 或 COBOL 那样严格规定书写格式。

(5) 每个语句和数据定义的最后必须有一个分号。分号是 C++ 语句的必要组成部分。

(6) C++ 语言本身没有输入/输出语句。输入和输出的操作是由“流”以及库函数来完成的。

这样做的目的,可以减小 C++ 语言本身的规模,也容易在各种机器上实现,提高了程序的可移植性。不同计算机系统所提供的函数个数和功能是有所不同的。

(7) 可以用//和/*……*/两种形式对 C++ 程序中的任何部分作注释。/*……*/是 C 语言中所用的注释方法,在 C++ 仍然可用,它主要用于多行的注释。注释只是为了增加程序的可读性,对编译和运行不起作用。/*……*/注释可以加在程序中空格可以出现的任何位置。而//是行注释,只能用在一行的末尾。//使用方便,但它只能用于 C++ 程序,不能用在 C 程序中。

一个好的有使用价值的源程序都应当加上必要的注释,以增加程序的可读性。

2. 从组织方式上看 C++ 程序的结构

从组织方式上看,C++ 程序的结构可归结如下:

(1) C++ 程序源程序由一个或多个文本文件组成。除用“# include”(包含)命令引入的文件外,C++ 的源文件是独立的编译的基本单位。

(2) 一个文件中可以包含一个或多个函数。如果将包含文件和被包含文件看做是一个文件,那么一个函数的定义不能分别写在两个或多个文件中。

(3) 在流行的 C++ 系统中,C++ 的源文件分为实现文件(文件名后缀为.cpp)和头文件(文件名后缀为.h)两类。以.c 为后缀的源文件也可以在 C++ 系统中使用,不过这时系统将它作为 C 的源程序看待,而不是作为 C++ 的源程序看待,因此,源程序中的 C++ 语句将被视为错误。

1.2.3 C++ 源程序的编辑、编译、链接和运行

1. 编辑

编辑就是在一定环境下进行程序的书写以及修改的过程。程序编码可以先写在纸上,也可以直接在计算机上用编辑软件编写,形成一个程序文件。

写出一个高级语言程序后,并不能立即执行,还必须进行编译和链接才能在机器上运行。

2. 编译

用户用程序设计语言编写的程序称为源程序,这样的程序,计算机是不能理解的。要计算机理解,必须要将源程序文件按该语言的词法和语法规则翻译成用二进制表示的程序文件——目标程序文件,这一过程称为编译。在编译过程中要对源程序的词法、语法和逻辑结构进行检查。编译工作是由称做编译器(Compiler)的软件完成的,不同的计算机程序设计语言有不同的编译器。

3. 链接

一方面,由于用户编写的程序可能被存放在多个源文件中,因此被编译生成多个目标程序文件(称为目标程序模块)。另一方面,几乎所有的程序都会使用到系统提供的一些目标程序模块(如执行输入/输出操作的模块)。因此,还必须将这些目标程序模块进行链接,才能生成一个完整的可执行程序。完成链接过程的软件称为链接器(linker)。显然,不同的程序设计语言提供有不同的链接器。图 1.4 为 C++ 语言程序的编辑、编译和链接过程示意图。

不少厂商为程序的开发提供了完善的集成开发环境,在集成开发环境中可以对源程序进行

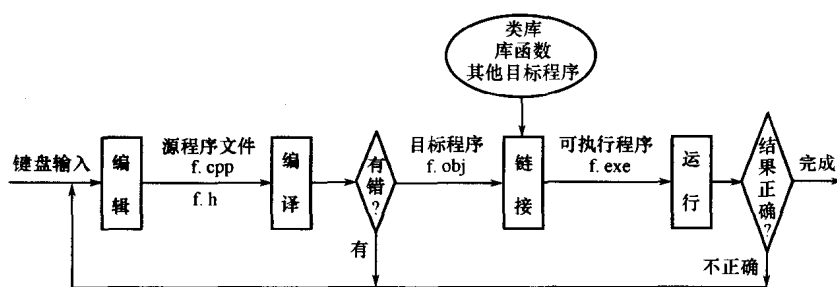


图 1.4 C++ 语言程序的编辑、编译和链接过程

编辑、编译、链接和运行。目前,可用于 C++ 的集成开发环境有 Turbo C++、Borland C++ 以及 Visual C++ 等。

1.3 C++ 集成开发环境的安装与运行

可用于 C++ 程序开发的系统很多,我们建议安装 Borland C++ 3.1 或 Visual C++ 6.0。

Borland C++ 3.1 内含面向 DOS 和 Windows 的两套开发系统,对于用过 Turbo C 系统的用户来说,Borland C++ 3.1 的安装和使用是非常简单和顺手的事,另外,它占用的空间也不大,还有一套简单实用的绘图系统,对于要求绘图的程序来说是一个最佳的选择。

Visual C++ 6.0 是一个面向对象、多窗口、能自动生成程序框架、功能强大、可视化的程序开发系统。在中文 Windows 下,它的编辑器能处理中文字符信息,开发出具有中文处理能力的 C++ 程序。

如果系统资源充足,可同时安装这两套系统,不会发生什么冲突,使用时各取所长。

由于 Borland C++ 3.1 的安装和使用比较简单,因此,下面主要介绍 Visual C++ 6.0 的安装和使用方法。

1.3.1 系统配置

对于 Visual C++ 6.0 来说,它要求的系统配置如下(这套配置对于 Borland C++ 3.1 来说是绰绰有余的):

- 配有 PentiumII 及其以上处理机的计算机
- 64 MB(或以上)内存
- 4.3 GB(或以上)的硬盘空间
- 一个 CD-ROM 驱动器
- Windows 9x、Windows NT 或 Windows 2000 操作系统

1.3.2 Visual C++ 6.0 集成开发环境的安装

Visual C++ 6.0 集成开发环境是 Visual Studio 6.0 软件包中的一个部分。Visual Studio 的安装

几乎是自动的,下面介绍几个主要的安装步骤:

(1) 在 Windows 环境下运行 Visual Studio 6.0 CD-ROM 上的 setup.exe 程序,将显示如图 1.5 所示的对话框。

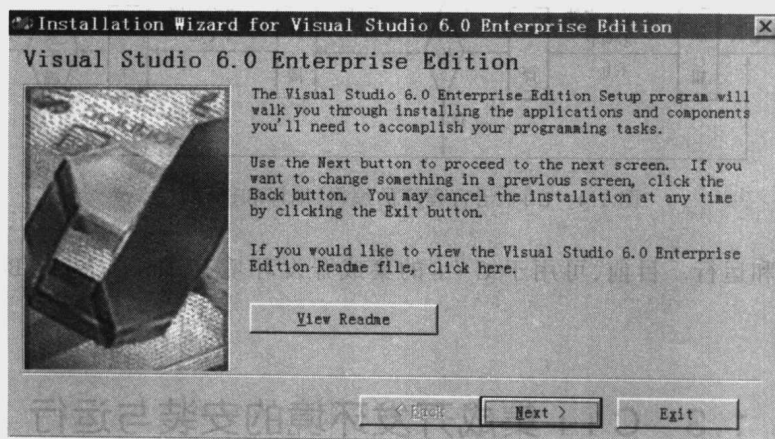


图 1.5 开始安装 Visual Studio

在这个对话框中安装向导告诉你一些简单的安装方法:单击“Next”按钮进入下一个屏幕;如果需要修改上一步中的某些项,可以单击“Back”键返回上一个屏幕;在任何时候如果想取消安装可单击“Exit”按钮。如果想看一看 Visual Studio 6.0 企业版的自述文件(Readme),可以单击“View Readme”按钮。

其后将是接受用户许可协议和填写产品登记号和用户信息的对话框,只有接受协议和正确填写产品登记号后才能进行其后的安装。

(2) 安装选择,如图 1.6 所示。

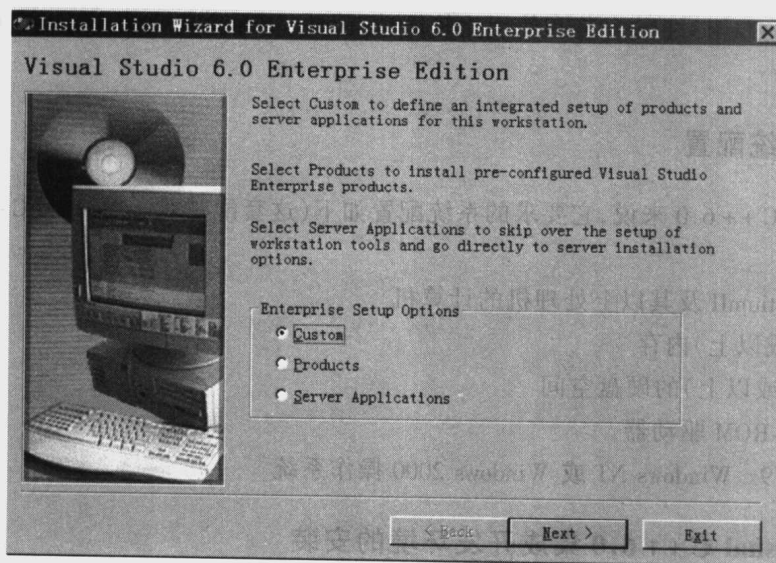


图 1.6 安装选择

保持默认选择,单击“Next”按钮,进入下一步。

(3) 在安装某个组件时,如果遇到已安装的同类产品,则会弹出如图 1.7 所示的对话框。

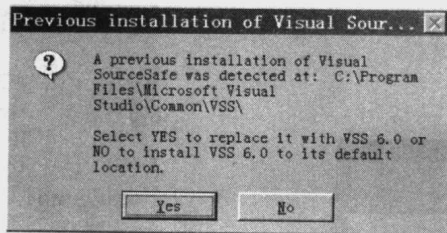


图 1.7 对新就版本组件的处理

此时,单击“Yes”按钮将以新的版本的组件置换老的版本,单击“No”按钮将保留老的版本,而新版本安装到默认的位置。

(4) 选择安装项目,如图 1.8 所示。

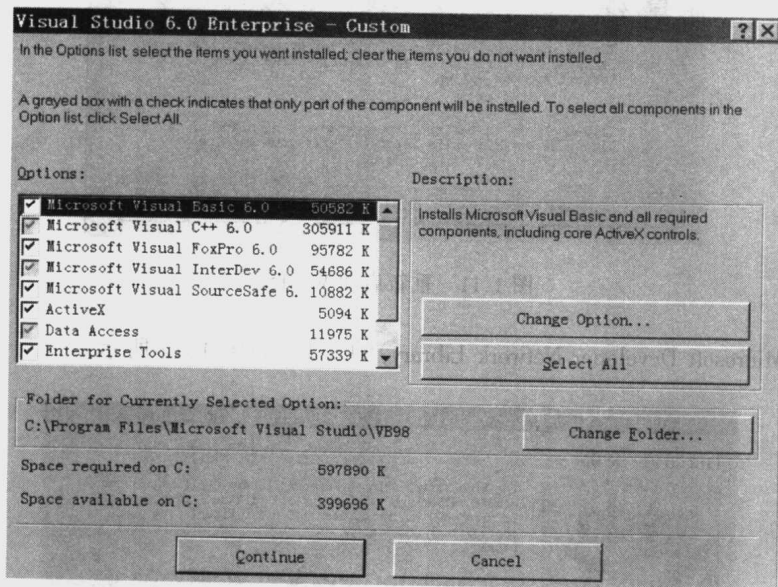


图 1.8 选择安装项目

在此,可以选择需要安装的项目,指定安装目录。对话框还报告了所需要的磁盘空间,所选的项目不同,所需要的磁盘空间也不一样。选定以后,单击“Continue”按钮进入下一步。

(5) 开始安装,系统显示安装进度,如图 1.9 所示。

(6) 若出现语言或代码页冲突,安装向导给出建议,如图 1.10 所示。

根据建议单击“是(Y)”按钮进行其后的安装。

(7) 安装过程要求重新启动系统,如图 1.11 所示。

单击“Restart Windows”按钮重新启动 Windows 并完成 Visual Studio 的安装和开始 MSDN 的安装。

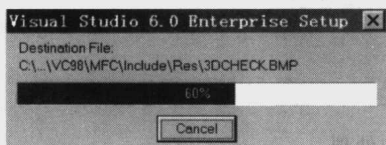


图 1.9 显示安装进度

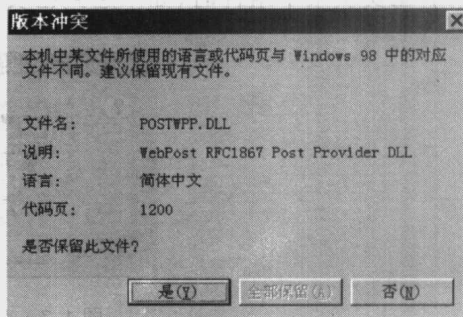


图 1.10 对语言或代码页冲突的建议

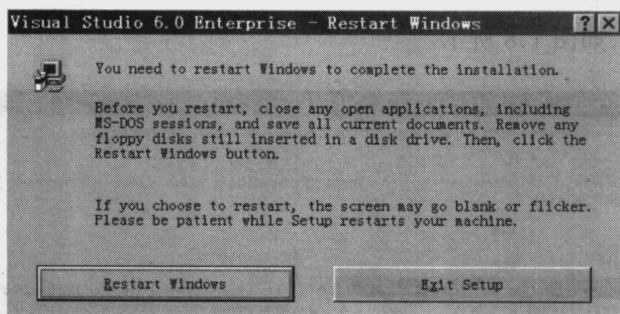


图 1.11 重新启动 Windows

(8) MSDN(Microsoft Developer Network Library)的安装,如图 1.12 所示。

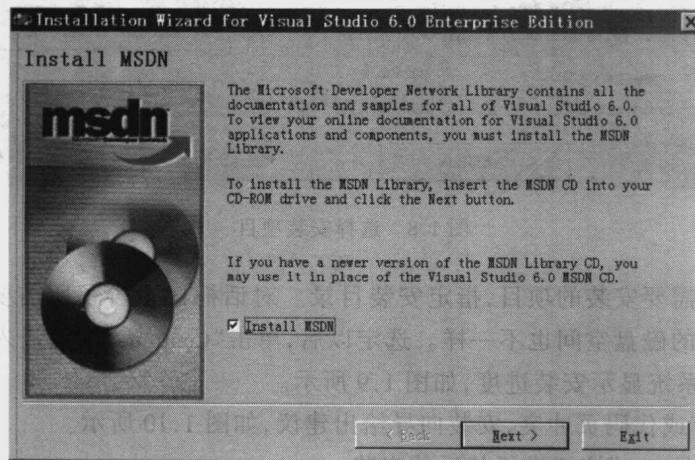


图 1.12 MSDN 的安装

MSDN 是 Microsoft Visual Studio 的公共联机文档,它使用户在编程时能够快速查询帮助信息,是学习和精通 Microsoft Visual C++ 的必备资料。