

Visual C++

系列丛书之一

Visual C++ 程序设计

秦勇 张克强 编著

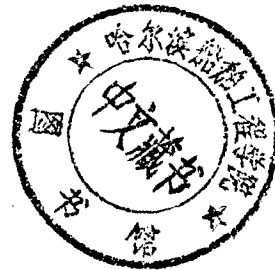


北京大学出版社



Visual C++ 程序设计

秦 勇 张克强 编著



北 京 大 学 出 版 社

新登字(京)159号

JS194/17

图书在版编目(CIP)数据

Visual C++程序设计/秦勇,张克强编著. —北京:
北京大学出版社,1994.6
ISBN 7-301-02510-6

I V...
I ①秦...②张...
II C语言-程序设计
IV TP312

出版者地址: 北京大学校内
邮政编码: 100871
排印者: 北京科技期刊出版集团电脑排印中心排版
北京飞达印刷厂印刷
发行者: 北京大学出版社
经销者: 新华书店
787×1092毫米 16开本 16.625印张 404千字
1994年6月第1版 1994年6月第1次印刷
印数: 0001—3500册
定价: 27.00元

前 言

本书主要讲述 C++ 语言中的一些基础知识,尤其是 C++ 有关类的知识是本书讲述的重点,在这部分详细讲述了类、封装、继承、多态性及虚拟函数等有关面向对象程序设计的方法和思想。另外类库也是 Visual C++ 一个重要组成部分,在本书也有部分章节对此进行了简略讲述。最后讲述了有关的一些编程技巧,如嵌入汇编、混合语言编程、内存模式选择等技巧性较强的知识和方法。本书以可移植性作为全书终结,表明可移植性是系统设计不可缺少的一部分。本书对于具有 C 语言知识和编程经验的读者学习 C++ 与面向对象程序设计是一本有益的教科书。

面向对象程序设计与面向对象程序设计语言是 90 年代才大量兴起并推动了软件开发的思想和方法。这种方法和思想一出现即引起计算机界、尤其是软件界的关注,广大计算机工作者都迫切希望了解这方面的知识、思想和方法。本书适应这一需要,借助 Visual C++ 这一重要的面向对象程序设计语言,介绍了面向对象程序设计的概念和方法。此外,本书也是读者学习 C++ 语言编程的重要工具书。本书例子丰富,有利于读者实际练习和掌握。因此,无论是对计算机专业的学生还是计算机工作者,本书均是一本值得一读的教科书。

下面介绍本书各章内容:

第一章 Visual C++ 的安装和使用。介绍 Visual C++ 安装、程序的编辑、编译及运行。

第二章 C++ 概述。介绍 C++ 区别于 C 语言的新特征。主要讲述标准输入、输出流设施、注释方法、原型说明方法、变量声明、作用域分辨符、枚举类型、const 关键字及 inline 函数、函数重载等概念。

第三章 引用。引用是 C++ 语言所提供的新的数据类型,它在传递函数参数和从函数返回值时起重要作用。

第四章 C++ 中的类基础。类是 C++ 中用户自定义数据类型的重要概念。本章讲述类的定义、类的构造函数、成员函数的定义方法,以及与类相关的信息封装、缺省构造函数、const 对象、对象生命周期等概念。

第五章 C++ 中类的使用。本章主要讲述类使用中的一些相关问题,如友元、静态成员、自由存储、copy 构造函数、this 指针、对象数组是本章讲述的重点。

第六章 类继承。继承是面向对象程序设计中的重要机制。本章介绍类派生的方法、派生类构造函数的编写方法、基类与派生类之间的转换及多重继承等概念。

第七章 虚拟函数与多态性。多态性是面向对象程序设计的一个重要特色,它是在继承的环境下建立在虚拟函数的概念之上的。本章还介绍了虚拟函数的实现方法——虚拟函数表,即 v-表,以及静态联编、动态联编和它们之间的相互区别,最后讲述纯虚拟函数、虚拟析构函数和虚拟基类等概念及编写方法。

第八章 运算符重载与转换函数。运算符重载和类型转换是面向对象程序设计的一个重要方面,是类继承环境的一个自然延伸。本章介绍重载运算符的方法及使用它的一些限制,最后讲述类型之间相互转换的方法。

第九章 Visual C++类库介绍。类库是 Visual C++的一个重要组成部分,一个面向对象程序设计语言类库设计的好坏决定了它是否使用方便,是否起到了代码重用的效果。本章从四个角度讲述了 Visual C++的类库,即通用类、程序结构类、OLE 类、可视对象类。Cobject 类是 Visual C++类库中大多数类借以派生的基类,在本章中也对其进行了介绍。

第十章 程序内存模式选择。Visual C++是设计 DOS 及 Windows 环境下软件的重要工具,二者运行的硬件环境是 PC 机及其兼容机,由于其特定的硬件结构,涉及到了程序内存管理模式的选择,本章重点介绍各种标准模式以及与 C++有关的模式方面的特有问題。

第十一章 嵌入汇编。汇编是局部优化和与硬件打交道的重要手段,在 Visual C++中可以嵌入汇编语句,本章重点介绍嵌入汇编的使用方法。

第十二章 混合语言编程。从利用已有程序代码、库和设计程序的方便等方面考虑,有时需要在 Visual C++程序中调用其它高级语言、模块或汇编语言模块,也可能是后者调用前者。本章讲述混合语言编程方面的一些协议,还讲述了 Visual C++与 BASIC、FORTRAN、PASCAL 相互调用的方法及混合语言编程中的数据交换方法。

第十三章 编写可移植 C++程序的一些考虑。可移植性和可读性、可维护性等都是设计软件时应考虑的重要问题。本章主要介绍编写可移植性软件在硬件及编译程序方面的一些考虑,以及数据文件的可移植性。

作者

1994 年 1 月

目 录

第一章	Visual C++ 的安装和使用	1
1.1	Visual C++ 的安装	1
1.1.1	进行 Visual C++ 安装之前的准备工作	1
1.1.2	系统的安装	2
1.2	Visual C++ 的使用	3
1.2.1	程序的编辑及保存	3
1.2.2	程序的编译及运行	4
第二章	C++ 概述	6
2.1	输出流	6
2.2	格式输出	7
2.3	标准错误输出流	8
2.4	标准输入流	8
2.5	C++ 语言程序中的程序注释方法	9
2.6	函数原型说明	9
2.7	变量声明	11
2.8	作用域分辨算符	12
2.9	枚举类型	13
2.10	const 修饰符	14
2.11	缺省函数参数	15
2.12	内联扩展函数	16
2.13	重载函数	17
2.14	链接说明	18
第三章	引用	21
3.1	引用的概念	21
3.2	初始化引用型变量	22
3.3	引用与指针的区别及相似点	22
3.4	引用作为函数参数	23
3.5	引用作为返回值	25
第四章	C++ 中的类基础	27
4.1	C 语言中结构是什么?	27
4.2	在 C++ 中如何声明类?	28
4.3	构造函数	29

4.4	析构函数	29
4.5	其它成员函数	30
4.6	声明对象和对象使用	31
4.7	public、private 和信息封装	32
4.8	缺省构造函数	34
4.9	成员函数与引用	35
4.10	const 对象	36
4.11	类嵌套	37
4.11.1	一个类嵌套例子	37
4.11.2	成员对象初始串列	38
4.11.3	const 对象成员	39
4.11.4	const 内部类型成员	39
4.12	对象的生命周期	40
第五章	C++ 中类的使用	43
5.1	友元	43
5.1.1	一个例子	43
5.1.2	友元类	44
5.1.3	友元函数	47
5.2	静态成员	47
5.2.1	一个例子	47
5.2.2	静态数据成员	49
5.2.3	静态成员函数	50
5.3	自由存储	51
5.3.1	自由存储	51
5.3.2	new 操作符	51
5.3.3	delete 操作符	52
5.3.4	内部类型与 new、delete 操作符	52
5.4	赋值操作符	53
5.4.1	一个例子	53
5.4.2	对象赋值	55
5.4.3	赋值操作符重载	55
5.5	copy 构造函数	57
5.5.1	copy 构造函数的背景	57
5.5.2	其它调用 copy 构造函数的场合	58
5.5.3	赋值操作符与 copy 构造函数的区别	59
5.6	this 指针	59
5.6.1	this 指针是什么?	60
5.6.2	this 指针与赋值操作符	60

5.6.3	使用 this 指针应注意的问题	62
5.7	对象数组	62
5.7.1	结构及结构数组	62
5.7.2	对象及对象数组	63
5.7.3	自由存储与对象数组	64
第六章	类继承	67
6.1	基类与派生类	67
6.1.1	C 语言中处理相关类的方法	68
6.1.2	C++ 语言中处理相关类的方法	69
6.1.3	派生类的定义方法	70
6.1.4	protected 关键字	71
6.1.5	类层次	72
6.2	基类及派生类的使用	73
6.2.1	公用基类与私用基类	73
6.2.2	资料隐藏和继承的关系	75
6.2.3	派生类中成员的重定义	76
6.3	派生类的构造函数	77
6.3.1	派生类构造函数定义	78
6.3.2	派生类构造函数的调用顺序	78
6.3.3	基类构造函数名称的省略	80
6.3.4	基类构造函数的定义	80
6.3.5	基类构造函数调用的省略	80
6.4	基类与派生类之间的转换	82
6.5	多重继承	84
6.5.1	多重继承的例子	84
6.5.2	多重继承下的派生类定义	84
6.5.3	多重继承下派生类构造函数的设计	85
第七章	虚拟函数与多态性	87
7.1	静态联编与动态联编	87
7.1.1	静态联编	87
7.1.2	动态联编	88
7.1.3	类型标识在静态联编下的使用	90
7.2	虚拟函数	94
7.2.1	虚拟函数的声明	94
7.2.2	虚拟函数的调用	96
7.2.3	对虚拟函数非动态联编方式的调用	97
7.2.4	虚拟函数的资料封装	99
7.2.5	怎样实现虚拟函数	101

7.3	纯虚拟函数	103
7.3.1	纯虚拟函数的定义	103
7.3.2	一个纯虚拟函数例子	103
7.3.3	使用纯虚拟函数的一些限制	107
7.4	虚拟析构函数	107
7.5	虚拟基类	110
7.5.1	虚拟基类定义的一般格式	111
7.5.2	虚拟基类继承关系下派生类的构造函数	112
第八章	运算符重载与转换函数	115
8.1	运算符重载	115
8.1.1	C语言中的运算符重载	115
8.1.2	C++语言中的一个数字类的例子	116
8.1.3	为Complex类定义重载运算符成员函数	118
8.1.4	重载运算符作为友元函数	121
8.2	运算符重载的注意事项	124
8.2.1	重载运算符的一些限制	125
8.2.2	重载运算符时必须遵守的原则	126
8.2.3	几点提示	127
8.3	重载下标运算符	128
8.4	重载new与delete操作符	130
8.4.1	重载new与delete操作符的方法	130
8.4.2	重载new操作符函数的附加函数	131
8.4.3	为类设计特殊的new与delete操作符重载函数	132
8.4.4	new、delete与构造函数、析构函数	134
8.5	类转换	135
8.5.1	C++类型转换应用的场合	135
8.5.2	基本数据类型与用户自定义数据类型的相互转换	137
8.5.3	用户自定义数据类型到其它数据类型的转换	139
8.5.4	二义性及其解决方法	141
第九章	Visual C++类库介绍	146
9.1	类库的一般介绍	146
9.1.1	Microsoft 基类库的特色	146
9.1.2	应用程序框架	147
9.2	通用类	149
9.2.1	内存管理	150
9.2.2	日期与时间	152
9.2.3	字符串	152
9.3	结构类	155

9.3.1	文档类	155
9.3.2	视窗类	157
9.3.3	用户接口类	159
9.3.4	应用程序类	160
9.4	OLE 类	160
9.4.1	OLE 的一些基本概念	161
9.4.2	OLE 类	162
9.4.3	客户程序的设计	163
9.4.4	设计服务程序	164
9.5	可视对象类	164
9.6	CObject 类介绍	164
9.6.1	由 CObject 派生新类的步骤	165
9.6.2	访问对象的运行时类信息	167
第十章	程序内存模式选择	169
10.1	标准模式介绍	169
10.1.1	指针	169
10.1.2	标准内存模式	171
10.1.3	小模式	172
10.1.4	微型模式	172
10.1.5	中模式	173
10.1.6	紧凑模式	173
10.1.7	大模式	175
10.1.8	巨型模式	175
10.1.9	空指针	176
10.1.10	标准模式的选择	177
10.2	混合模式	177
10.2.1	变量定义	178
10.2.2	near 函数和 far 函数	179
10.2.3	指针转换	180
10.3	自定义内存模式	181
10.3.1	代码指针大小的设定	181
10.3.2	数据指针大小的设定	182
10.3.3	段设置	182
10.4	based 指针	187
10.4.1	based 指针	187
10.4.2	基址数据的分配	190
10.5	based 地址与函数	192
10.6	虚拟存贮管理	193

10.7	类的内存模式	193
10.7.1	环绕内存模式	194
10.7.2	修改环绕内存模式	194
10.7.3	重载 this 指针	195
10.7.4	指定返回对象的寻址方式	196
10.7.5	虚拟函数表指针	197
10.8	new、delete 与内存模式	197
10.8.1	new 操作符	197
10.8.2	delete 操作符	198
10.8.3	_set_new_handler 函数	199
10.9	成员函数的基址寻址	199
第十一章	嵌入汇编	201
11.1	嵌入汇编介绍	201
11.2	--asm 关键字	201
11.3	嵌入汇编使用方法	202
11.3.1	嵌入汇编语言的特色	202
11.3.2	--asm 块中 C 或 C++ 语言成分的使用	204
11.3.3	寄存器的使用方法	207
11.3.4	浮点数指令的使用方法	208
11.3.5	标号的使用方法	208
11.3.6	C 或 C++ 函数的调用	209
11.3.7	定义--asm 块作为宏	210
11.3.8	程序优化	211
第十二章	混合语言编程	212
12.1	混合语言编程的协议	212
12.1.1	关于命名规则方面的要求	212
12.1.2	关于调用规则方面的要求	213
12.1.3	关于参数传递规则方面的要求	214
12.1.4	程序的编译和链接	215
12.2	C 语言程序调用其它高级语言	216
12.2.1	C 调用 Basic	217
12.2.2	C 调用 FORTRAN	218
12.2.3	C 调用 Pascal	220
12.3	C 语言程序调用汇编语言	222
12.3.1	过程的设置	223
12.3.2	进入过程	224
12.3.3	分配局部数据	224
12.3.4	保存寄存器	225

12.3.5	存取参数	225
12.3.6	返回值	227
12.3.7	退出过程	228
12.4	C++语言程序调用其它高级语言	228
12.5	混合语言编程中的数据处理	229
12.5.1	缺省的命名及调用协议	229
12.5.2	数字数据表示方法	230
12.5.3	字符串	230
12.5.4	数组	233
12.5.5	数组的声明及数组下标	234
12.5.6	结构、记录和用户定义数据类型	235
12.5.7	外部数据	235
12.5.8	指针和地址变量	236
12.5.9	公共块	236
12.5.10	可变数的参数	238
第十三章	编写可移植 C++ 程序的一些考虑	239
13.1	硬件与可移植性	239
13.1.1	基本数据类型及其存储位数	239
13.1.2	结构或联合中各个域的存储顺序及其对齐方式	242
13.1.3	算术运算模式	246
13.1.4	指针	246
13.1.5	字符集	248
13.2	编译程序与可移植性	248
13.2.1	符号扩展	249
13.2.2	右移位操作	250
13.2.3	标识符的长度和大小写	250
13.2.4	寄存器变量	251
13.2.5	可变个数参数的函数	251
13.2.6	求值顺序	252
13.2.7	带副作用的函数或宏的参数	252
13.2.8	运行环境的差别	253
13.3	数据文件的可移植性	253
13.4	Visual C++ 中专门的可移植性考虑	253
13.5	Visual C++ 中的字节顺序	253

第一章 Visual C++的安装和使用

本章作为全书的开篇章,将首先介绍 Microsoft Visual C++ 的安装,然后介绍 Visual C++ 程序的编辑、编译及运行,以便读者在学习本书的过程中随时将书中的例子程序录入计算机,编译和运行它,并验证程序的运行结果,从而更好地理解书中所讲述的概念、思想和方法。

1.1 Visual C++ 的安装

Visual C++ 系统的安装非常简单,但在进行安装之前,必须做好两件准备工作,才能顺利地安装好 Visual C++ 系统。

1.1.1 进行 Visual C++ 安装之前的准备工作

Visual C++ 系统的大部分应用程序都运行于 Microsoft 公司的 Windows 环境下,更主要的是 Visual C++ 的安装程序 Setup 也必须在 Windows 环境下运行,所以在安装 Visual C++ 系统之前,必须首先在硬盘上安装 Microsoft 公司的 Windows 操作系统,否则就不能进行该系统的安装。

安装 Visual C++ 系统所要做的第一件准备工作是为系统安装准备足够的硬盘空间,否则可能出现下列情况:系统安装一部分时您被告知系统没有足够的硬盘空间继续进行安装,这时您必须退出安装程序,为系统重新准备更多的硬盘空间,然后重新开始安装。这可能使您几分钟,甚至几十分钟的安装工作前功尽弃,给您带来不愉快。

系统安装到底需要多少硬盘空间,取决于您在“安装选项”应答窗中(Installation Options)所做的选择。例如您可以选择不安装例子程序,从而减少系统安装所需要的硬盘空间。在 1.1.2 小节中将讲述如何选择安装系统的某一部分或全部。

如果您选择安装系统的全部,Visual C++ 的安装程序 Setup 在安装过程中需要 45,442KB 的硬盘空间。系统安装完成以后,系统实际所需要的硬盘空间为 42,650KB。部分安装因所做的安装选择不同,所需要的硬盘空间也会有所差异,在安装过程中安装程序 Setup 将根据您所做的安装选择告诉您实际需要的硬盘空间。尽管如此,安装之前预先了解以上的两个数据是很有帮助的。

由于 Visual C++ 安装过程中将拷贝大量的文件,所以安装 Visual C++ 系统所要做的第二件准备工作包含两个方面:

其一,必须保证系统配置文件(CONFIG.SYS)中所设定的缓存区数目大于或等于 30(即在 CONFIG.SYS 文件中必须对 BUFFERS 进行设定,且所设定的数目必须大于或等于 30)。

其二,必须保证由哪个软驱安装 Visual C++ 系统,该软驱就必须被设置为能够为磁盘数据交换进行超高速存储缓冲。

如果以上两方面不能得到保证,那么 Visual C++ 系统的安装将非常慢,在某些系统上安装时间可能超过一小时。

1.1.2 系统的安装

Visual C++ 系统的安装非常简单,其中最关键的是“安装选项”应答窗(Installation Options),在应答窗中有 8 个检测框(Checkbox),分别是:

- ① Visual Workbench;
- ② C/C++ Compiler(C/C++ 编译器);
- ③ Foundation Class(基类库);
- ④ AppStudio : Resource Editor(资源编辑器);
- ⑤ Runtime Libraries(运行库);
- ⑥ Sample Code(例子程序);
- ⑦ Online Help Files(在线帮助文件);
- ⑧ Tools(工具)。

这 8 个检测框分别设定是否安装 Visual Workbench、C/C++ 编译器、基类库、AppStudio 的资源编辑器、运行库、例子程序、在线(Online)帮助文件及各种工具。如果检测框中有×标记,则表示对应项将被安装。否则对应项将不被安装。

在应答窗中还有 4 个比较关键的按钮,这 4 个按钮分别是:目录(Directories)、库(Libraries)、帮助文件(Help Files)及例子程序(Samples),点中这 4 个按钮将分别弹出 4 个新的应答窗,分别是:

- ① 目录设定应答窗;
- ② 库安装设定应答窗;
- ③ 帮助文件安装设定应答窗;
- ④ 例子程序安装设定应答窗。

在“目录设定应答窗”中可以设定系统将安装在哪个硬盘驱动器以及安装在哪个目录。

在“库安装设定应答窗”中可以选择安装哪种内存模式的运行库及选择将安装的数学库。

在“帮助文件安装设定应答窗”中可以选择安装哪个方面的帮助文件,其中有 4 类帮助文件,分别是:类库帮助文件、C 语言或运行库的帮助文件、Windows 3.1 SDK(Windows 3.1 开发工具)帮助文件以及笔和多媒体(Pen、Multimedia)方面的帮助文件。

在“例子程序安装设定应答窗”中可以选择安装用户指南例子程序、Windows C++ 基类库例子程序、Windows SDK(Windows 开发工具)例子程序。

“安装选项”应答窗(Installation Options)设定完之后,就可以按照安装程序的提示,一张一张地插入软盘进行安装。

1.2 Visual C++的使用

这里所讲的 Visual C++ 的使用不是讲如何使用 Visual C++ 语言,而是讲述如何使用 Visual C++ 编辑、编译及运行程序。

1.2.1 程序的编辑及保存

Visual C++ 系统安装好之后,Visual C++ 的安装程序 Setup 将在 Windows 下建立一个 Microsoft Visual C++ 组(Group),由 Program Manager 即可进入该组(图 1.1)。

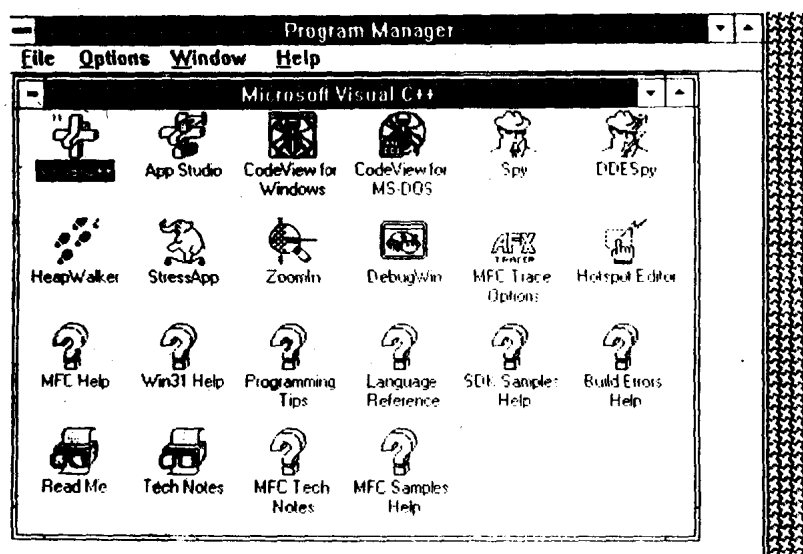


图 1.1

由 Program Manager 启动 Visual Workbench(在图 1.1 所示屏幕显示下连点两下 Visual C++ 图标,即可进入 Visual Workbench 应用程序),之后读者即可编辑及保存文件了。读者可以试着录入如下简短的程序:

```
#include <iostream.h>
void main()
{
    cout << "Hello, you're welcome\n";
}
```

现在屏幕显示可能类似于图 1.2 所示。

选择 File 菜单中的 Save As 命令(或与该命令对应的图标),并给程序取名为 HELLO.CPP 以保存前面所录入的程序。现在屏幕显示可能类似于图 1.3。

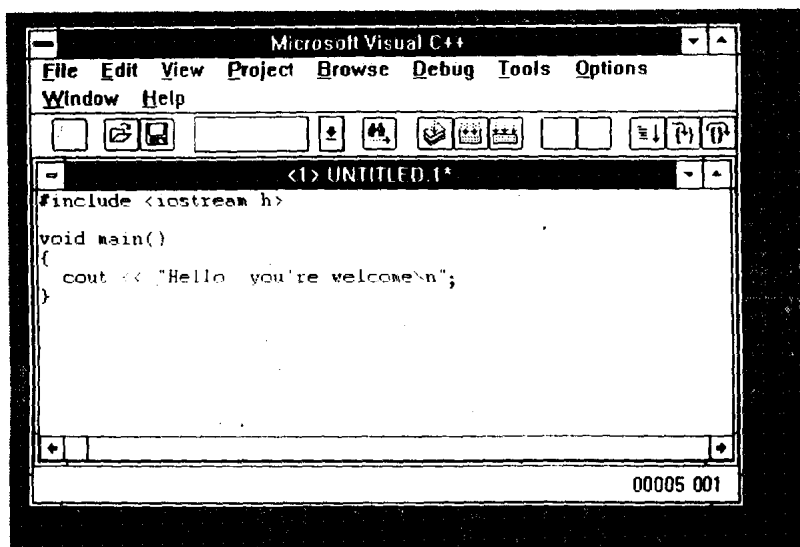


图 1.2

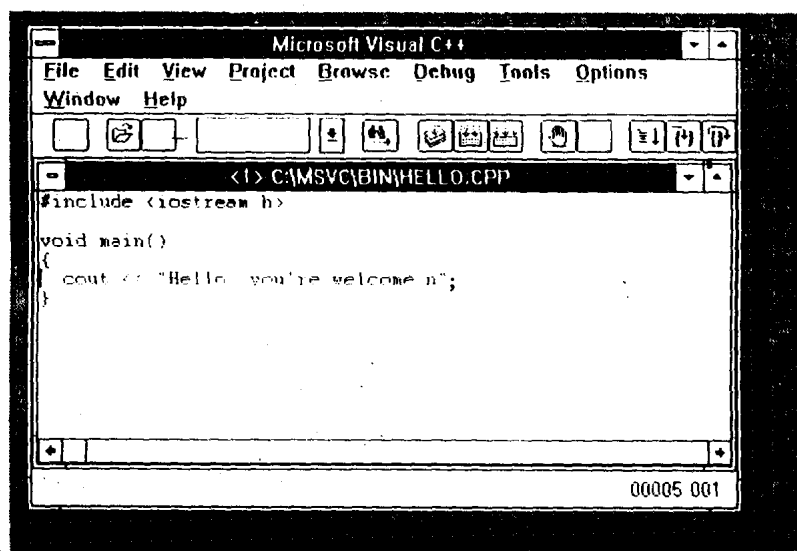


图 1.3

1.2.2 程序的编译及运行

选择 Project 菜单中的编译及链接命令(或与该命令对应的图标),这时屏幕显示可能类似于图 1.4。

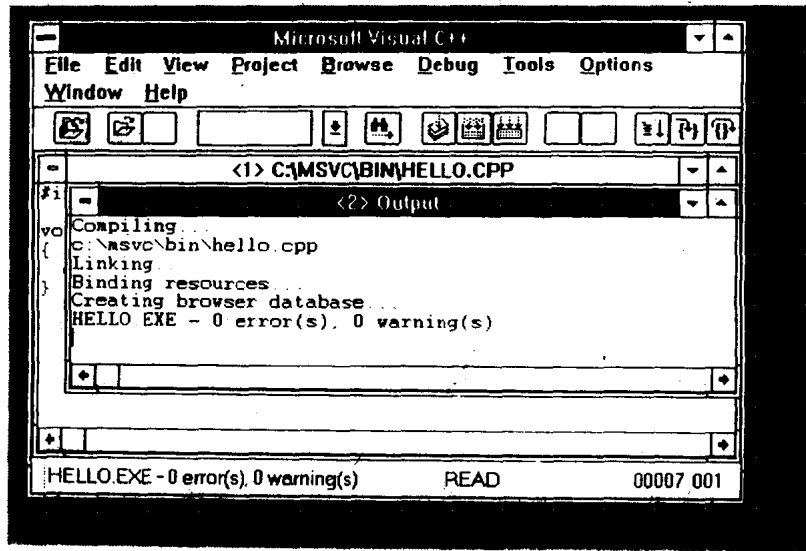


图 1.4

选择 Project 菜单中的运行程序命令(或与该命令对应的图标),这时屏幕显示可能类似于图 1.5。

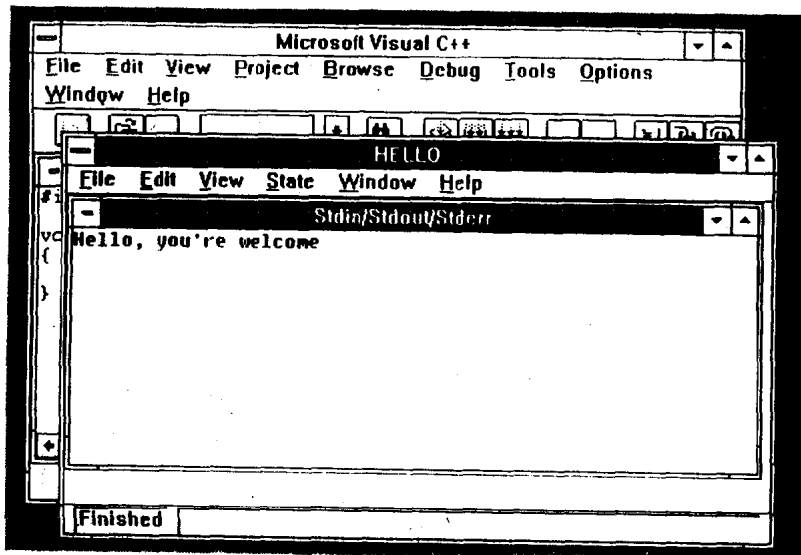


图 1.5

这就是前面所录入程序——HELLO.CPP 的执行结果。