

高等学校计算机课程规划教材

C++程序设计实践教程

高福成 主编 边莫英 主审

高等学校计算机基础课程规划教材

C++ 程序设计实践教程

高福成 主编

边奠英 主审

高福成 潘旭华 宁爱军 王春娴 编

内容提要

本书是普通高等教育“十一五”国家级规划教材《C++ 程序设计教程》的配套教材,本着面向应用、注重实用、读者好用的原则,为学习 C++ 程序设计的读者上机实习和自我测试安排了大量的练习题和测试题。书中内容由浅入深、循序渐进,既要掌握 C++ 语言的基础知识,也要强调对计算机算法的理解和程序设计思维方法的介绍,同时注重创新思维能力的培养;既着眼普及也兼顾提高,可适合不同层次读者的需要。

本书可作为高等学校 C++ 程序设计课程的教学参考书,也可供工程技术人员自学使用。

图书在版编目(CIP)数据

C++ 程序设计实践教程/高福成主编. —天津:天津大学出版社,2009. 1

ISBN 978-7-5618-2860-1

I. C... II. 高... 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(2008)第 190607 号

出版发行	天津大学出版社
出版人	杨欢
地 址	天津市卫津路 92 号天津大学内(邮编:300072)
电 话	发行部:022-27403647 邮购部:022-27402742
网 址	www.tjup.com
印 刷	天津泰宇印务有限公司
经 销	全国各地新华书店
开 本	185mm × 260mm
印 张	14.75
字 数	405 千
版 次	2009 年 1 月第 1 版
印 次	2009 年 1 月第 1 次
印 数	1 - 2 000
定 价	25.00 元

凡购本书,如有缺页、倒页、脱页等质量问题,烦请向我社发行部门联系调换

版权所有 侵权必究

前 言

本书是普通高等教育“十一五”国家级规划教材《C++ 程序设计教程》的配套教材。该书本着面向应用、注重实用、读者好用的原则,为学习 C++ 程序设计的读者提供上机实习指导和自我测试练习题。

在程序设计方法中,面向对象的程序设计方法是现代程序设计的里程碑,是面向过程结构化程序设计方法的基础。作为一个合格的程序设计者,既要掌握结构化程序设计方法,又要掌握面向对象的程序设计方法。本书实习指导部分以 1998 年国际标准化组织(ISO)颁布的 C++ 语言的国际标准 ISO/IEC 14882—1998 为依据,以最新的集成开发环境 Visual Studio. net 2005 为工具,引导读者由浅入深地进行结构化程序设计到面向对象程序设计的上机训练。其中,前五章为面向过程的结构化程序设计的训练,可以帮助读者提高结构化程序设计的能力;后四章为面向对象程序设计的训练,使读者初步掌握面向对象编程的能力。这部分练习题包括两种题型:一种给出了求解问题的算法分析或编程思路,同时给出了源代码,目的是使初学者能在模仿中学习和掌握解决某一类问题的程序设计的思维方法;另一种则需要由读者自行编程,从而充分发挥自己的潜能和积累编程的经验。按照先模仿编程然后自主编程的学习方法,既能克服初学者对程序设计的无从下手的畏难情绪,也能拓展他们自我发挥的空间,有利于读者创新能力的发挥。

自我测试题部分按 C++ 语言的主要知识点编排了较大数量的实用性较强的题目,供读者进行自我检查和考核。这部分内容的目的是培养初学者程序设计的基本功,即通过选择题考核语言主要知识点的熟悉程度,通过阅读程序题测试阅读现有程序的能力,通过程序填空题检测根据已有算法编制程序的能力,从而掌握程序设计的思维方法和提高编程技巧。

本书由高福成主编,边奠英主审。参加编写的有潘旭华、宁爱军和王春娴。由于作者的水平所限,书中不妥之处,敬请读者指正。

编者

2008 年 12 月

目 录

第 1 篇	Visual Studio. net 2005 集成开发环境的使用	(1)
第 1 章	用 VS 2005 开发 C++ 应用程序的过程	(1)
1.1	单源文件控制台应用程序的开发	(2)
1.2	打开已有的项目	(7)
1.3	多源文件控制台应用程序的开发	(8)
1.4	创建头文件的方法	(11)
第 2 章	应用程序的安装和卸载	(16)
2.1	应用程序的安装和卸载	(16)
2.2	程序的安装	(18)
2.3	卸载	(20)
第 2 篇	上机指导	(21)
实验 1	数据类型和数据运算	(21)
实验 2	程序控制结构	(24)
实验 3	复合数据类型	(35)
实验 4	函数	(42)
实验 5	指针与引用	(48)
实验 6	类和对象	(57)
实验 7	继承和派生	(68)
实验 8	多态性和虚函数	(74)
实验 9	模板	(80)
实验 10	综合程序设计	(86)
附录	上机指导参考程序	(87)
第 3 篇	练习题	(137)
第 1 章	数据类型和数据运算	(137)
第 2 章	程序控制结构	(141)
第 3 章	复合数据类型	(154)
第 4 章	函数	(164)
第 5 章	指针与引用	(171)
第 6 章	类和对象	(180)
第 7 章	继承和派生	(190)
第 8 章	多态性和虚函数	(202)
第 9 章	模板	(214)
附录	练习题答案	(223)

第 1 篇 Visual Studio. net 2005 集成开发环境的使用

因为 C++ 是一门相当复杂的语言,所以编译器也难于构建,完全符合 C++ 标准的编译器仍很少见,表 1.1 列出了几种与标准 C++ 兼容性较好的常见的集成开发环境。鉴于 Visual Studio. net 具有人们较为熟悉的运行界面和较大的兼容性,以及容易为读者接受等优势,本书选择 Visual Studio. net 2005(简称 VS 2005)作为教学示范环境。

表 1.1 常见的 C++ 集成开发环境

集成环境	与标准 C++ 的兼容性	操作系统平台	特点
Borland C++ builderX 2.0	100%	Windows	支持 C++98 标准
Visual Studio. net	98.22% 以上	Windows	支持 C++98 标准,兼容 VB、C#、J#
Dev-C++	96.15%	Windows	Windows 下的 GCC 版本
GCC 4.0(G++)	96.15%	Linux、UNIX	兼容 Java、FORTRAN
Visual C++ 6.0	83.43%	Windows	不支持 C++98 标准

第 1 章 用 VS 2005 开发 C++ 应用程序的过程

在 VS 2005 中,一个 C++ 应用程序被称为一个“项目”(Project)。项目是由应用程序中需要的所有文件组成的一个有机整体,一般包括源文件、头文件、资源文件等。项目能自动将其包含的文件进行分类和管理,从而大大减轻程序员的负担。

项目被置于“解决方案”(Solution)的管理之下。一般情况下,每个项目或多个项目都需要共同的环境支持,这种环境支持称为解决方案。一个解决方案可以包含多个项目,除了应用程序项目外,可能还有一些静态库项目(lib)和一些动态库项目(dll),还可能会使用一些第三方库。这些项目相互独立,但共用一个解决方案的环境设置。解决方案设置成一个目录结构并配合一定的环境变量,这样不但可以使得代码管理更井井有条,项目成员之间也更容易相互配合,更重要的是能够使得最终应用程序的安装包制作、源代码打包发布和转移变得十分容易。

在 VS 2005 中开发一个 C++ 控制台应用程序的步骤如下:

- ①启动 VS 2005 集成开发环境;
- ②创建新项目,必要时可同时创建解决方案;
- ③在项目中创建和编辑源文件、头文件等程序文件(或将这些文件添加到项目中);
- ④对源程序进行编译和连接,生成可执行的目标文件;

⑤执行应用程序。

1.1 单源文件控制台应用程序的开发

下面介绍在 VS 2005 平台上进行 C++ 单源文件控制台应用程序开发的过程。

1. 启动 Visual Studio 2005

选择“开始”→“程序”→“Microsoft Visual Studio 2005”→“Microsoft Visual Studio 2005”或双击桌面快捷方式图标,可以启动 Visual Studio 2005,显示如图 1.1 所示的 Visual Studio 2005 主窗口,即“起始页”窗口。由于初始设置不同,看到的窗口可能与图 1.1 有些差别。

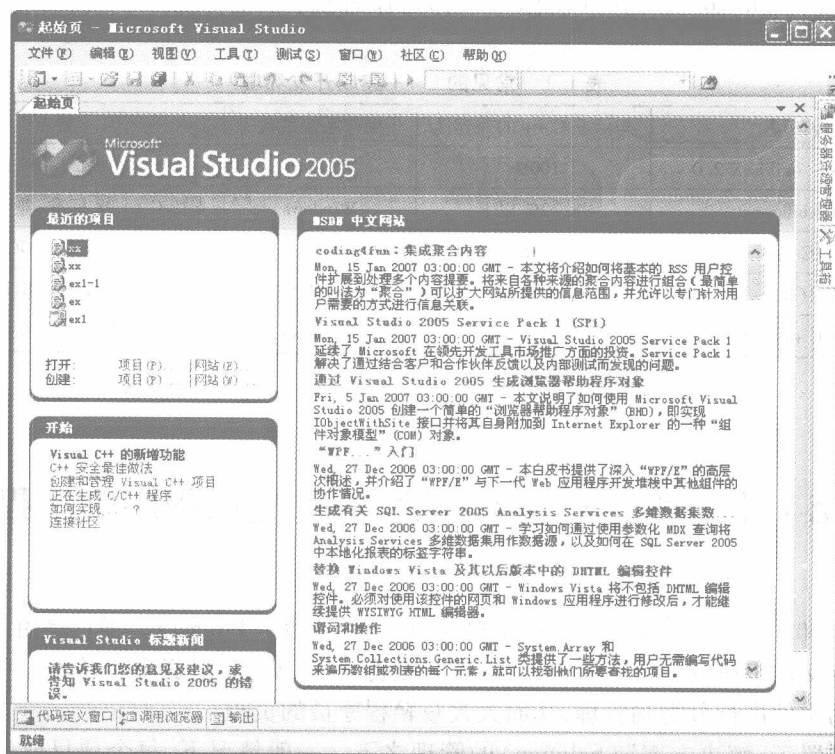


图 1.1 Visual Studio 2005 “起始页”窗口

“起始页”窗口用于访问项目、阅读产品新闻和访问 MSDN,以使用户更有效地利用 Visual Studio。起始页一般由四个独立的信息区域组成。

1) “最近的项目”区域 用于打开已经创建的项目或创建新项目。区域上部显示用户已经创建的项目名称,如图 1.1 中“ex”和“ex1”等,单击选定的项目可以直接打开它;下部是“打开”和“创建”两个命令,相当于主菜单中的“文件”→“新建”/“打开”功能。如果要创建一个新的应用程序,单击“创建”右边的“项目(P)”命令;如果要创建一个新的 Web 网站应用程序,单击“创建”右边的“网站(W)”命令。

2) “开始”区域 用于提供面向初学者的联机帮助信息,如 Visual C++ 的新增功能、C++ 安全最佳做法等。

3) “Visual Studio 标题新闻”区域 可以连接到 MSDN 产品反馈信息网站,便于用户向该网站反馈使用本产品的意见或建议。

4) “MSDN 中文网站”区域 可以连接到 MSDN 网站,直接阅读该网站提供的产品信息。

2. 创建新项目和解决方案

①执行“文件”→“新建”→“项目”命令,弹出图 1.2“新建项目”对话框。

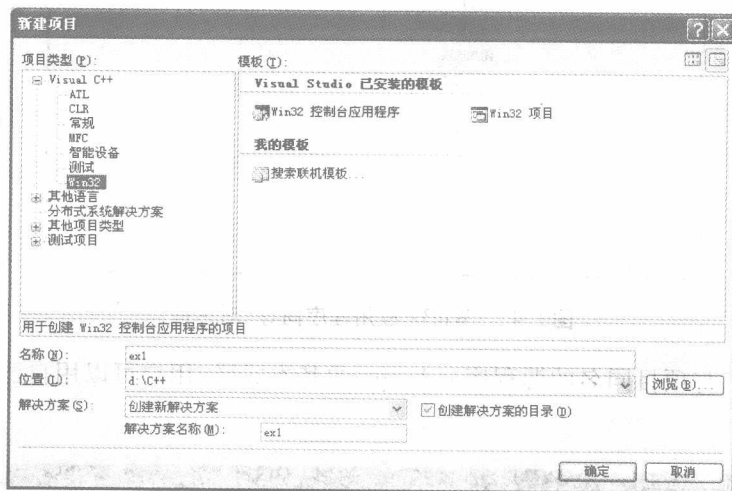


图 1.2 “新建项目”对话框

②在对话框左侧的“项目类型”中选择“Win32”,在右侧的“模板”中选择“Win32 控制台应用程序”,在“名称”文本框中输入用户自己命名的项目名称,如 ex1,并选择适当的存储位置,如 d:\C++ (C++ 文件夹应提前建立),并在“解决方案”复选框中选择“创建新解决方案”。这样,VS 2005 在创建新项目时,会创建一个与项目的名称相同的解决方案。因此,当在“名称”文本框中输入项目名称时,在“解决方案名称”文本框中将自动出现与项目相同的名称。最后单击“确定”按钮,弹出“Win32 应用程序向导”对话框,如图 1.3 所示。单击“下一步”按钮,进入下一个“应用程序设置”对话框(如图 1.4),在“附加选项”中选中“预编译头”,目的是自动生成含有预编译命令的应用程序设计模板,然后单击“完成”按钮。

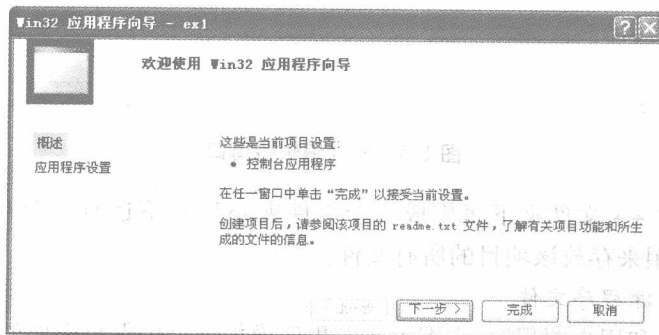


图 1.3 Win32 应用程序向导(概述)

③选定的应用程序设计模板自动生成一个模板程序,并进入图 1.5 所示编辑器窗口。

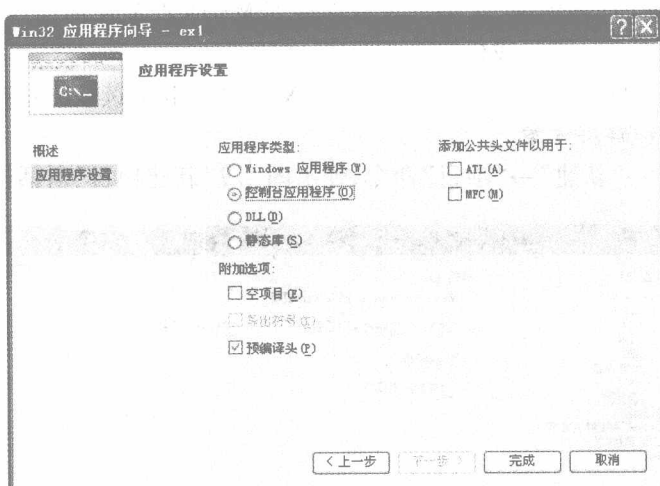


图 1.4 “Win32 应用程序向导”对话框

程序模板给出了与项目同名的源程序 ex1.cpp 的基本式样,用户可以用自己的源程序修改它。

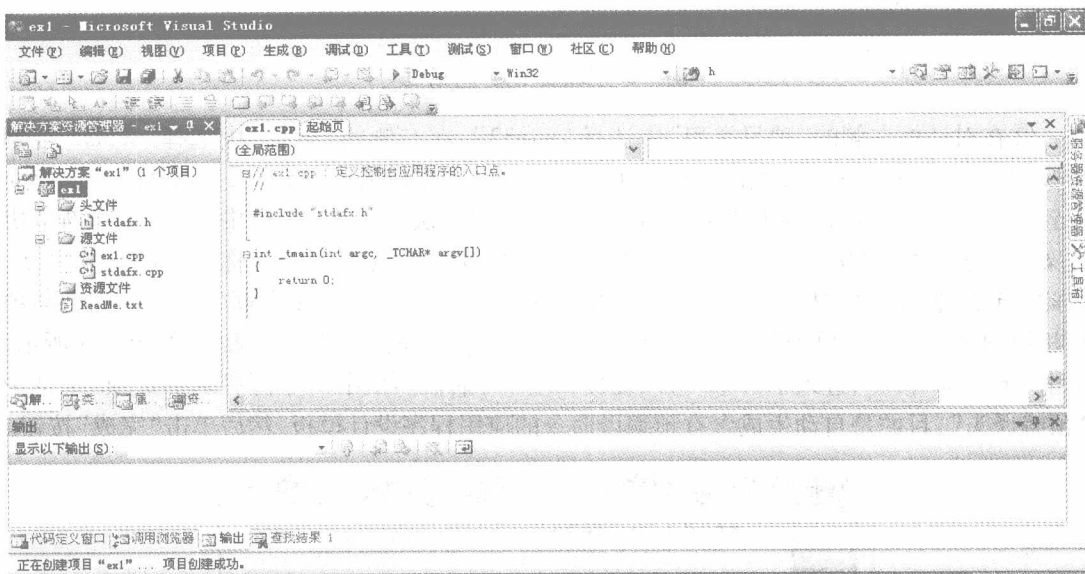


图 1.5 文件编辑器窗口

至此,在 d:\C++ 文件夹下将生成一个文件夹 ex1,其下还有一个同名的项目文件夹 ex1。项目文件夹用来存放该项目的文件。

3. 创建和编辑源程序文件

对照模板程序和用户源程序,在图 1.5 的程序模板窗口中保留需要的部分,再输入用户源程序的其他部分,就产生了自己的源程序,如图 1.6 所示。将该程序与例 1.1 的程序比较后发现,程序的第 2 行增加了一个#include "stdafx.h"命令。这是 C++ 新标准提供的头文件

预编译,就是把一个项目中使用的一些标准头文件预先编译,以后编译该项目时,就不再编译这部分头文件,仅仅使用预编译的结果,从而可以加快编译速度,节省时间。

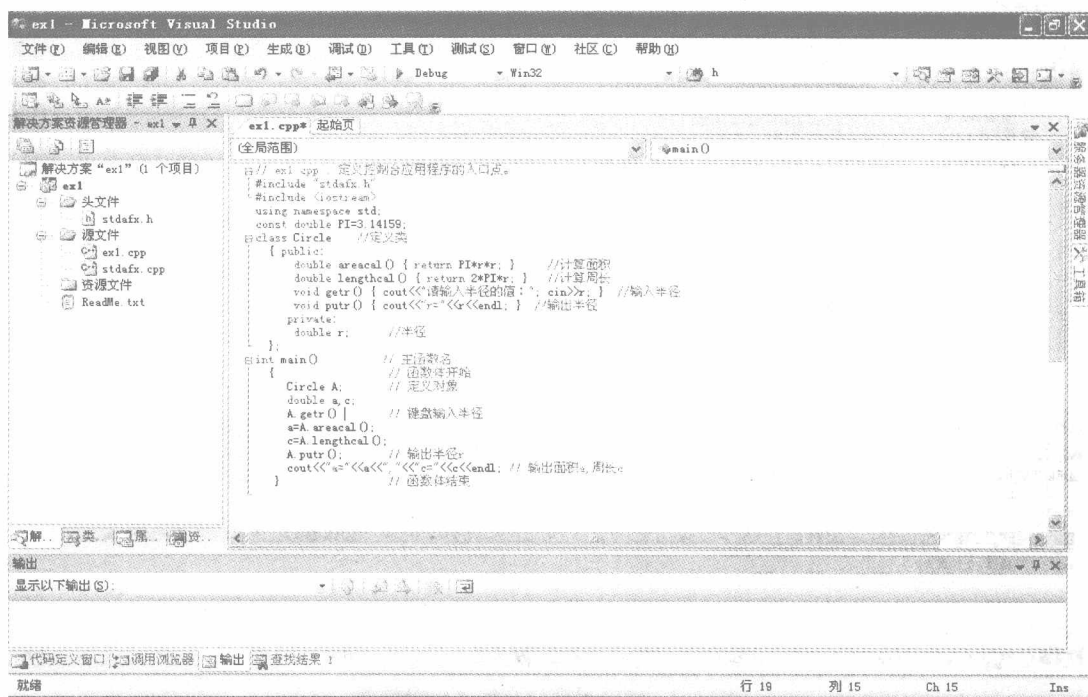


图 1.6 在编辑器窗口中编辑源程序

4. 编译和连接

对编辑完成后的源程序,用主菜单“生成”下的“生成 ex1”命令进行编译和连接。

若在此过程中发现错误,将在编译窗口显示源文件,在输出窗口显示出错信息,指出在哪一行出现了什么性质的错误,如图 1.7 所示。

图中的错误是:d:\c++\ex1\ex1\ex1.cpp(20) : error C2146:语法错误:缺少“;”(在标识符“a”的前面)。这说明在源程序第 20 行发现编号为 c2146 的语法错误,该错误的原因是标识符 a 前面缺少“;”,因此编译不成功。双击该出错信息行,在编辑窗口第 20 行左侧出现一个箭头指向错误行,经检查发现,问题出在上一行末尾少了语句结束符“;”。这时,可以在第 19 行末尾添加“;”,然后重新编译。显示图 1.8 所示编译器窗口及编译成功信息。

在编译过程中,可以通过“调试”菜单进行异常处理、跟踪程序的运行、逐语句调试、在程序中设置新断点或删除程序中的所有断点等操作,以便更有效地进行程序调试。

编译完成后,将在 d:\C++\ex1 和 d:\C++\ex1\ex1 下各自生成一个 debug 文件夹,前者存放可执行文件 ex1.exe,后者存放 stdafx.h 生成的预编译头文件 ex1.pch、预编译类型文件 stdafx.obj、源程序编译生成的目标文件 ex1.obj 等,在 d:\C++\ex1 文件夹下还有一个特殊文件 ex1.sln,这是一个环境文件。当打开一个已有项目时,要用到这个文件。

5. 运行

编译和连接成功后,可以用主菜单“调试”下的“开始执行(不调试)”命令多次运行程序而不必重复编译和连接,运行结果如图 1.9 所示。

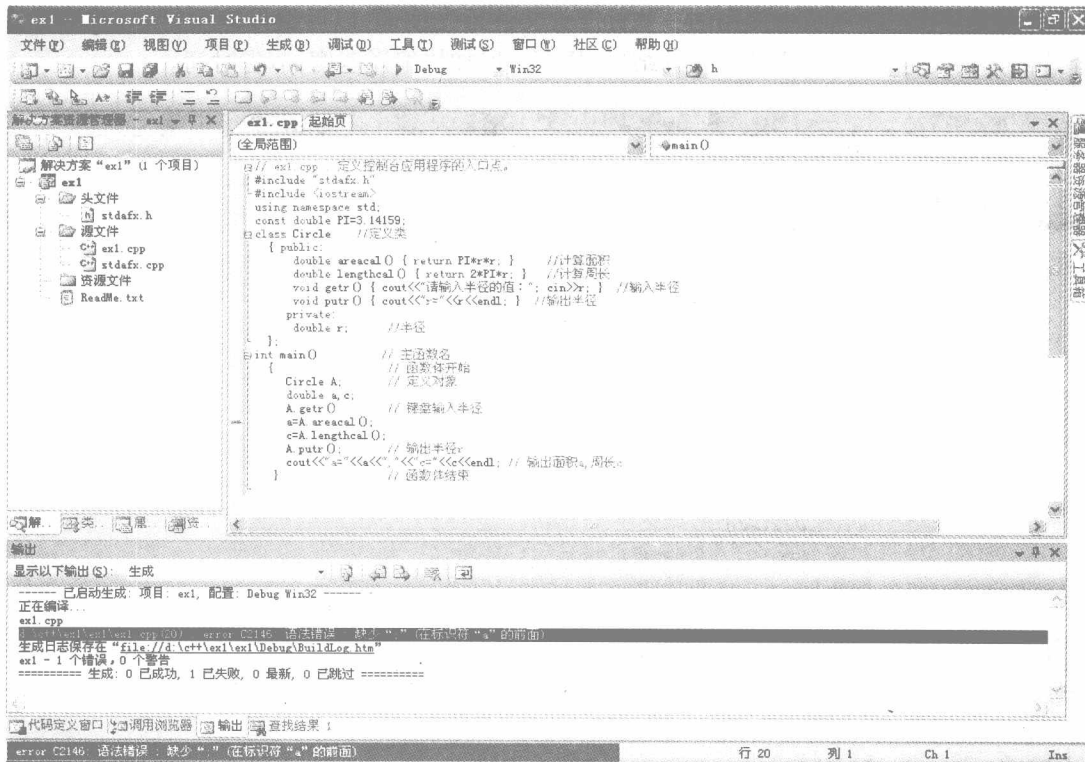


图 1.7 编译器窗口及编译出错信息

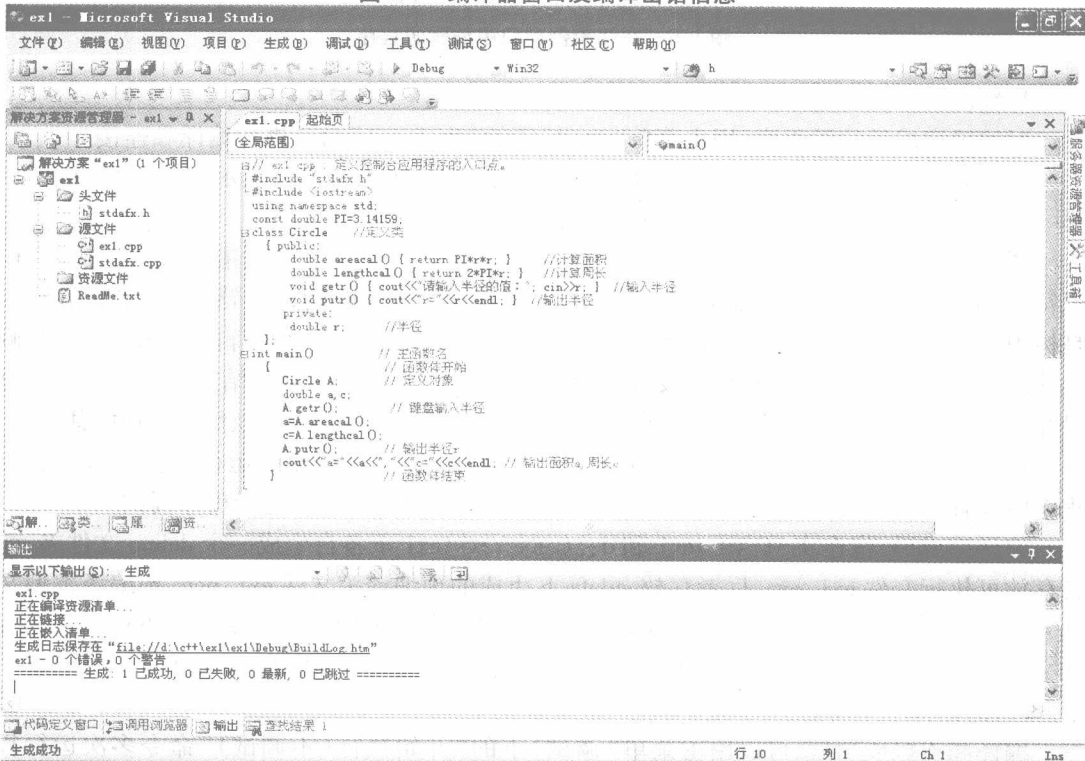


图 1.8 编译窗口和编译成功信息

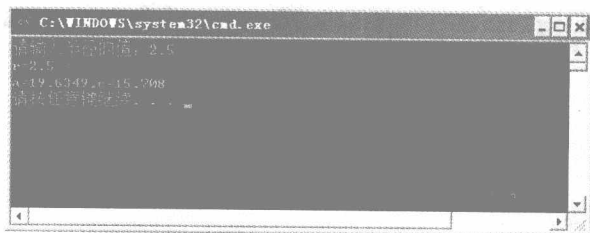


图 1.9 程序运行结果

1.2 打开已有的项目

重新启动 VS 2005 后,若要打开一个已有的项目,应选择“文件”→“打开”→“项目/解决方案”,弹出图 1.10“打开项目”对话框。

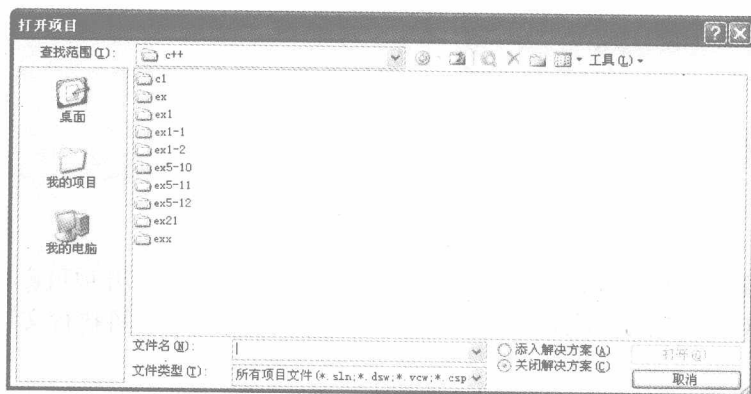


图 1.10 “打开项目”对话框(解决方案文件夹)

该对话框中将显示已经创建的各个解决方案文件夹,如 cl、ex 等。若双击其中的 ex1 文件头,弹出图 1.11 对话框。从对话框中选择环境文件 ex1.sln 并打开它,即可打开指定项目,并显示源程序编辑窗口,如图 1.12 所示。在该窗口中可进行原有程序的编辑、编译、连接和运行。

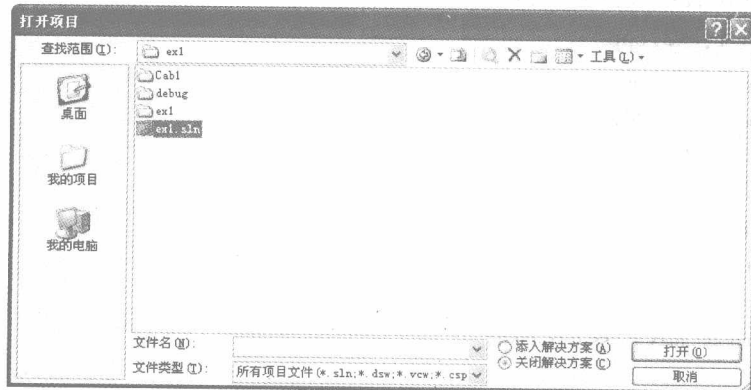


图 1.11 “打开项目”对话框

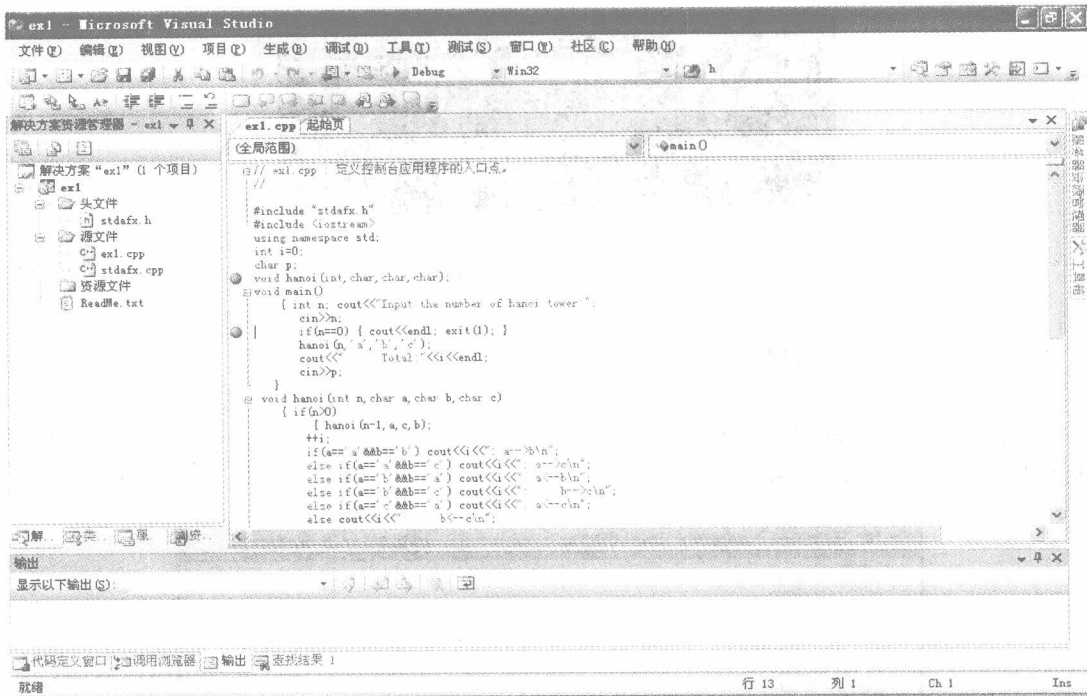


图 1.12 打开已有项目窗口

如果只想重新运行已编译和连接成功的可执行程序,就不必打开项目窗口,直接在资源管理器中打开对应解决方案文件夹下的 debug 文件夹,找到其中的可执行文件,双击它即可运行。

1.3 多源文件控制台应用程序的开发

如果一个项目包含多个源文件,则需要先创建含有 `main()` 函数的主源文件,然后创建其他源文件,并将这些文件逐个添加到项目中,最后生成和运行可执行的目标程序。

下面的程序由程序文件 `ex2` 和 `ex2-1` 组成。

```
//文件 1:ex2. cpp
#include <iostream>
using namespace std;
int a=5,b=6; //定义全局变量 a,b
extern int max(int x,int y);
int main()
{ int c;
  c = max(a,b);
  cout << "max =" << c << endl;
}

//文件 2:ex2-1. cpp
int max(int x,int y)
```

```

{
    int z;
    z = x > y ? x : y;
    return z;
}

```

```
static int a = 100, b = 70; //定义静态全局变量 a, b
```

为创建上述含多个源文件的项目,操作方法如下。

①按照创建单源文件的方法创建一个项目(项目名为 ex2),然后新建源文件(默认文件名为 ex2.cpp),该文件必须包含主函数 main()。编辑完成后,该文件自动保存在 ex2\ex2 文件夹(即项目文件夹)下,如图 1.13 所示。

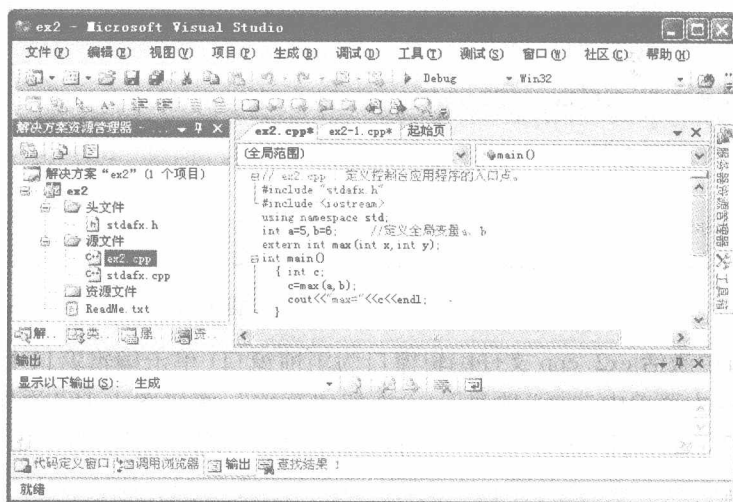


图 1.13 源文件 ex2.cpp 编辑窗口

②执行主菜单中“文件”→“新建”→“文件”命令,弹出“新建文件”对话框,如图 1.14 所示。从对话框左侧的“类别”中选择“Visual C++”,从右侧的“模板”中选择“C++ 文件”,单击“打开”按钮,弹出新的程序编辑窗口,默认名为“源 1.cpp”。在编辑窗口输入 ex2-2.cpp 的内容,注意程序开头应增加一行:

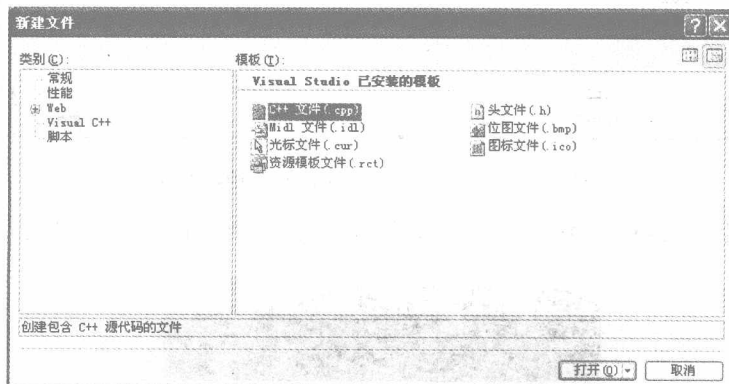


图 1.14 “新建文件”对话框


```
#include "stdafx.h"
```

如图 1.15 所示。编辑完成后,用“文件”→“源 1.cpp 另存为”命令将源 1.cpp 更名为 ex2 - 1.cpp 保存到项目文件夹下。

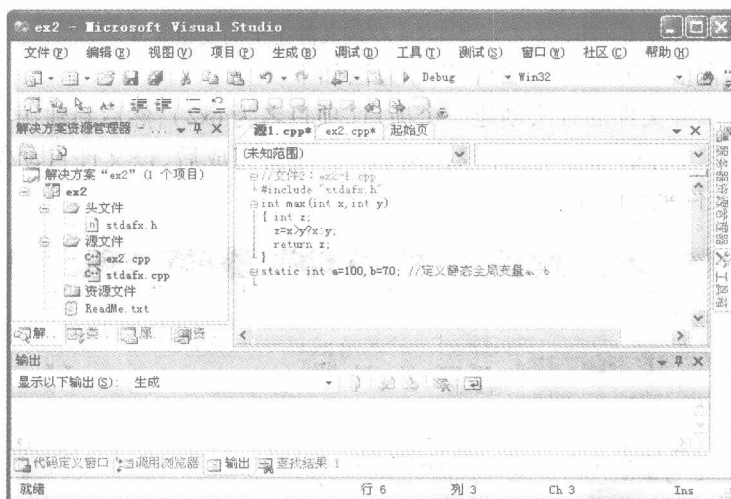


图 1.15 源文件 ex2 - 1.cpp 编辑窗口

③在主窗口中激活 ex2.cpp 文件编辑窗口为当前窗口(单击编辑窗口上部标题栏中的 ex2.cpp *),并在主菜单中选择“项目”→“添加现有项”,弹出图 1.16“添加现有项”对话框,从中选择 ex2\ex2 文件夹中的 ex2 - 1.cpp,然后单击“添加”按钮,将 ex2 - 1.cpp 添加到项目 ex2 中。

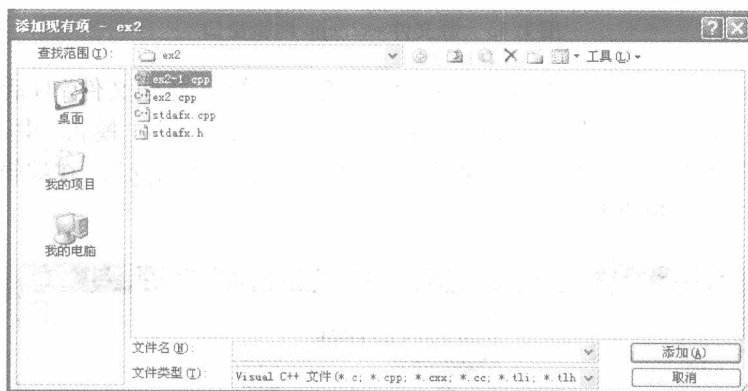


图 1.16 将 ex2 - 1.cpp 添加到当前项目中

④系统返回 VS 2005 主窗口,对 ex2 进行编译、连接和运行,结果如图 1.17 所示。



图 1.17 项目 ex2 运行结果

1.4 创建头文件的方法

一个复杂的 C++ 项目经常要定义各种类,定义或使用各种函数模板和类模板。在这种情况下,通常将定义的和模板存放在不同的头文件中,以保证它们的全局特性。本节介绍如何创建用户自行定义的头文件的操作方法。

为了在 VS 2005 环境中创建用户自定义的头文件,首先要创建一个与头文件同名的源文件,该源文件只有两条编译预处理命令,即

```
#include "stdafx.h"
#include "头文件"
```

其中第 2 行为用户自定义的头文件。

创建用户自定义头文件的操作步骤如下:

- ①创建项目及含有 main() 函数的主源文件;
- ②创建头文件及与头文件同名的源文件;
- ③将头文件及与头文件同名的源文件添加到项目中;
- ④生成和运行可执行目标文件。

下面的程序由三部分组成:源文件 ex3.cpp、用户头文件 looplist.h 和与头文件同名的源文件 looplist.cpp。程序如下:

```
//looplist.h                                //将模板定义设置为头文件
template <typename T>
class Node                                    //节点类模板定义
{ public:
    Node<T> * next;                          //指向下一个节点的指针成员
    T data;                                  //本节点数据成员
};

template <typename T>
class LoopList                               //链表类模板定义
{ public:
    LoopList()                               //构造函数定义
    { tail = head = NULL; }
    void Append(T x);                       //成员函数声明
    void MinSum();                          //成员函数声明
    ~LoopList();                           //析构函数声明
    Node<T> * tail, * head;                //tail 存放链尾节点,head 存放链头节点
};

template <typename T>
void LoopList<T>::Append(T x)               //成员函数:向链尾添加节点
{ Node<T> * p;
    if ((p = new Node<T>) == NULL)         //向堆内存申请一个节点的内存空间
```

```

        { cout << "allocation failed!"; exit(1); }
    p->data = x; //将数据存入新节点 data 域
    p->next = head; //新节点 next 域指向头节点
    if ( head == NULL) //是否空链
        head = tail = p; //链中生成第一个节点
    else
        { tail->next = p; tail = p; } //在尾部添加新节点
}

template < class T >
void LoopList < T > :: MinSum ( ) //成员函数:求和及最小值
{
    Node < T > * p = head;
    T m3, m;
    m = m3 = p->data + p->next->data + p->next->next->data;
    cout << m3 << "\t"; //输出前三个节点数据之和
    for ( p = p->next; p != head; p = p->next )
        { m3 = p->data + p->next->data + p->next->next->data;
          cout << m3 << "\t"; //输出相邻三个节点数据之和
          if ( m > m3 ) m = m3;
        }
    cout << "MIN = " << m << endl; //输出最小值
}

template < class T >
LoopList < T > :: ~LoopList ( ) //析构函数:删除链表中所有节点
{
    Node < T > * p;
    while( tail != head )
        { p = tail;
          tail = tail->next;
          delete p;
        }
}

//looplist. cpp //与头文件同名的源文件
#include "stdafx. h"
#include "looplist. h"

//ex3. cpp //主源文件
#include < iostream >
using namespace std;
#include "looplist. h"
const int N = 5;
void main ( )

```