
北京化工大学

# Beijing University of Chemical Technology

## 面向对象程序设计 (CSE24312C)

---

# 上机实验指导书

---

**Cheng Yong**

## 目录

创建日期：2020 年 8 月 10 日
更新日期：2020 年 8 月 12 日

# 前言

1946 年世界上第一台数字电子计算机 ENIAC 诞生，标志着人类进入信息时代。六十年来，计算机科学取得了飞速迅猛的发展，计算机及其应用系统已经渗透到社会生活的方方面面，有力地推动了整个社会的进步。

随着软件规模的扩大和复杂性程度的提高，传统的面向过程编程语言如 C、Basic、Pascal、COBOL 等的缺点逐渐显露，其中最主要的就是数据与过程的分离。数据和过程在这些语言中都是相互独立的实体，这不仅加重了程序员的编程负担，而且客观上也与人们分析和解决问题的思维方式不一致。这促使人们寻找一种更为“自然”的语言来描述和求解问题。

1967 年，挪威计算中心的 Ole-Johan Dahl 和 Kristen Nygaard 共同开发了 Simula 语言，命名为 Simula-67，首次提出了对象 (Object) 的概念以及将数据和操作进行封装的思想，尽管 Simula-67 中所实现的对象还不太成熟，但它却为面向对象语言的发展奠定了基础。鉴于 Ole-Johan Dahl 和 Kristen Nygaard 在面向对象语言设计方面的奠基性工作，他们被授予 2001 年度 ACM 图灵奖。

1971-1975 年，Xerox PARC 实验室 Alan Kay 领导的小组实现了 Smalltalk 语言，这是第一个全面实现面向对象思想的语言，在程序设计语言历史上具有重要的里程碑意义。Smalltalk 不仅引入子类和继承的概念，全面实现了面向对象思想，而且还开发了第一个集成开发环境 (IDE) 和应用开发环境 (ADE)。对后来的 Objective-C、Actor、Java 和 Ruby 等语言产生了深远的影响。1980 年，Smalltalk-80 推出后，面向对象技术逐步成为主流编程语言，Alan Kay 也因此被授予 2003 年度 ACM 图灵奖。

目前，面向对象的系统分析、设计和实现已经成为软件工程的主流开发方法论。本课程旨在讲授在实践开发中广泛应用的 C++ 语言，主要内容包括面向对象的基本思想，类、对象、继承和多态等基本概念及其在 C++ 语言中的实现，此外还包括异常处理、运算符重载、输入输出处理、标准模板库等内容，为了更深入地帮助学生理解和掌握这些知识点，本课程提供了 12 学时的上机实践课时，分六次实验 (每次 2 个学时)，所列如下：

(1) C/C++ 语言基础知识；

(2) 类与对象

(3) 运算符重载

(4) 继承与派生

(5) 虚函数与多态性

(6) 输入输出处理

上述实验对于理解 C++ 语言的基本知识点是必不可少的。实践证明：多上机实践，多做项目是提高学生编程技巧的重要途径。所有实验都假定在以下环境下进行：

(1) 操作系统：Windows 2000/XP/Vista；

(2) 开发环境：Visual Studio 6；

如需要更改到其他环境，请参考其他操作系统和开发环境的相关资料，以便顺利进行上述实验。为了确保上机实验效果，要求在实验中遵循以下要求：

(1) 实验前认真预习实验内容和要求；

(2) 根据自身知识的掌握情况，选定实验内容；

(3) 上机实验过程中，完成编码工作；

(4) 对编写的程序进行认真的分析，发现存在的问题，并进行测试；

(5) 独立撰写严谨的、条理通顺的、字迹端正的实验报告；

# 实验 1　C/C++ 语言基础

## 1.1　实验目的

(1) 熟悉使用 Visual Studio 6 来编写和调试程序；

(2) 理解数组、指针和引用；

(3) 掌握数组声明、数组初始化、数组作为函数参数、字符串数组等；

(4) 掌握指针声明、指针初始化；

## 1.2　实验要求

(1) 学会使用 Visual Studio 6 环境，掌握编辑、编译和运行 C++ 程序的完整过程；

(2) 认真复习 C 语言基础知识；

(3) 在 Visual Studio6 环境下完成选做实验内容；

(4) 认真撰写完整的实验报告；

## 1.3 实验内容

(1) 熟悉 Visual Studio 6 集成开发环境，了解菜单和工具栏的意义，学会创建控制台项目，并掌握如何编辑、编译、链接和执行 C++ 程序，并学会初步调试 C++ 程序。

(2) 输入一个正整数，判断它是否为回文数，所谓回文数即正读和反读都是一样的数，如 12321、56465 都是回文数。

(3) 约瑟夫 (Josephus) 问题：$n$ 个人围坐成一圈，从 1 开始顺序编号；游戏开始，从第一个人开始由 1 到 $m$ 循环报数，报到 $m$ 的人退出圈外，问最后留下的那个人原来的序号。

(4) 定义一个二维数组，数组维度由用户输入，通过函数求数组的鞍点 (二维数组的鞍点就是该位置上的元素在该行上是最大而在该列上是最小，数组可能没有鞍点)，函数参数分别为指针和引用的传址方式；

## 1.4 实验步骤

(1) 首先，确保实验机器上是否安装 Visual Studio6，如果没有，或者选择安装或选用其他的开发环境；

(2) 参考相关资料和文档，掌握 Visual Studio6 集成开发环境的使用方法和 C++ 程序的编辑、编译、链接、运行的完整过程；

(3) 根据自身掌握知识的情况，选作实验内容，并完成编码和调试；

## 1.5 实验提示

目前，支持标准 C++ 的编译器很多，如微软、Borland 等公司和 GNU 等软件组织都推出了功能相当强大的 C++ 编译器。由于 Windows 操作系统的广泛使用，这里主要介绍微软公司推出的企业级集成开发环境 Visual Studio，最新版本为 2005。考虑到机房版本为 6，故这里主要介绍 Visual Studio 6。包括以下内容：

(1) 掌握 Visual C++ 集成开发环境的使用，如进入和退出，菜单以及工具栏，用户窗口区的划分 (workspace, scource file, message)；

(2) 程序的编辑、编译、链接和运行方法；

(3) 初步的程序调试方法；

## 1.6 Visual C++ 集成开发环境简介

启动并进入 Visual C++ 集成开发环境有三种方法：

(1) 在开始菜单上，选择程序，然后 Microsoft Visual Studio 6.0 组，再选择 Microsoft Visual C++6.0，如图 (1.1)。

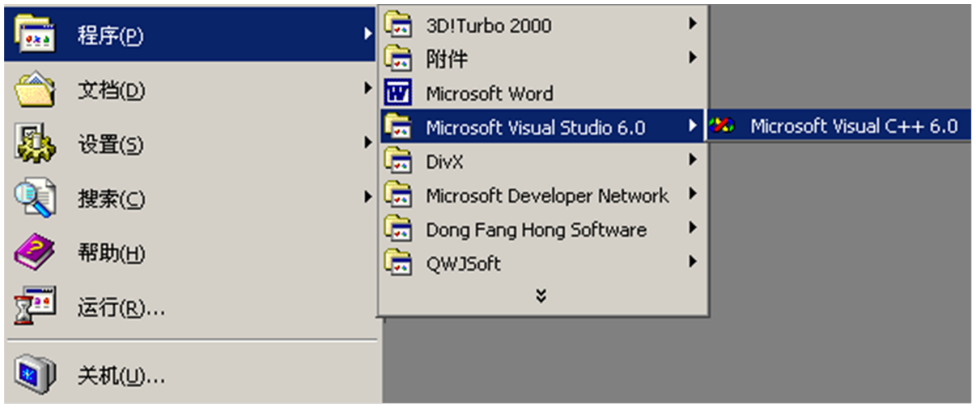


图 1.1: 启动 Visual C++ 集成开发环境

(2) 在桌面上创建 Microsoft Visual C++6.0 的快捷方式，直接双击该图标；

(3) 如果已经创建了 VC 的某种工程，双击该工程的 dsw(Develop Studio Workshop) 文件图标，也可进入集成开发环境，并打开该工程；

启动 Visual C++ 之后，就可以看到如图 (1.2) 所示的主界面：

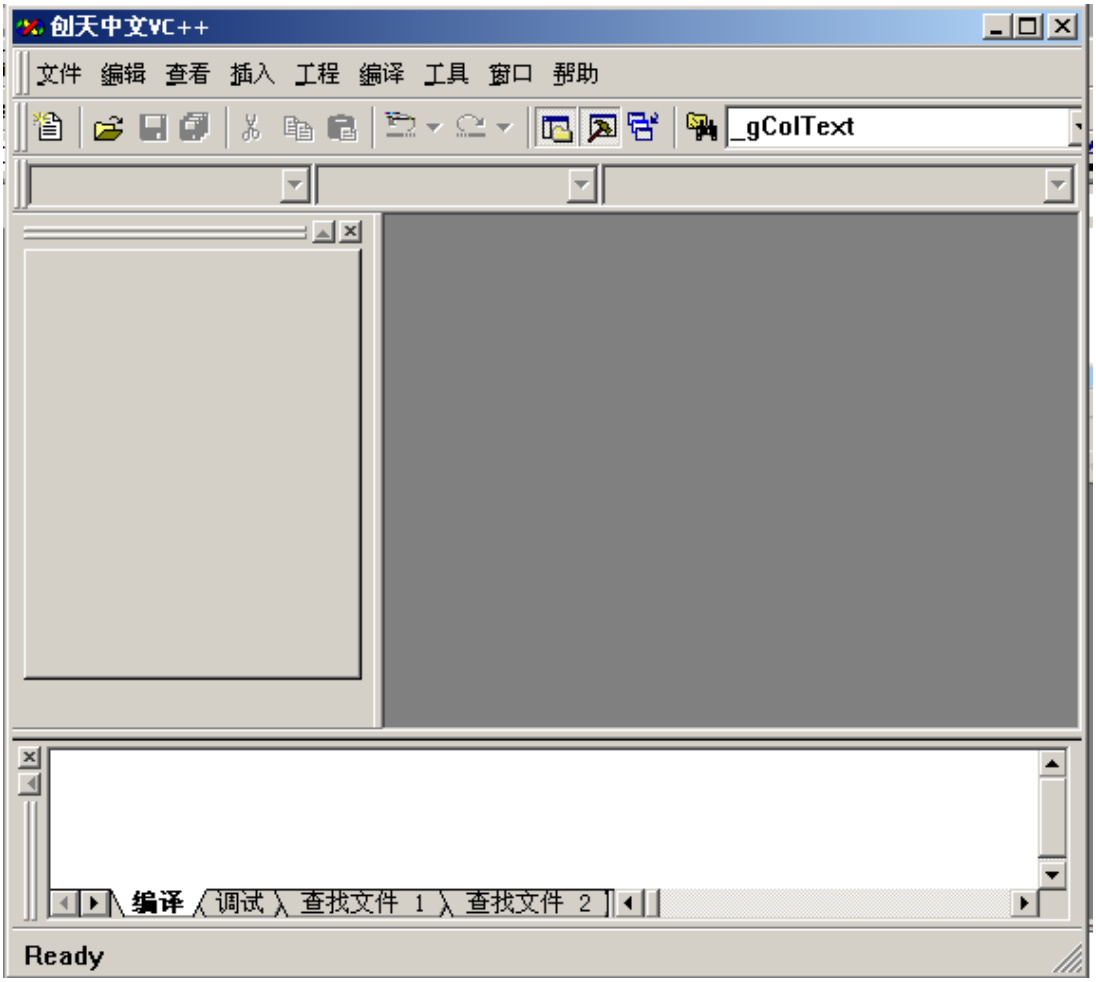


图 1.2: Visual C++ 集成开发环境主界面

关于主界面上菜单、工具栏的详细介绍，可以参考帮助说明，这里不再详细介绍。如要退出集成开发环境，可选择 File 菜单下的 Exit 命令。

**控制台应用程序**

下面以一个简单的例子，来学习了解如何利用 Visual C++ 来进行简单的控制台应用程序开发。

**例 1.** *要求用户从键盘输入 3 个整数，然后按照从小到大的顺序在屏幕上输出。*

**项目创建**

(1) 首先在资源管理器中，在用户盘 (硬盘) 创建自己的文件夹，如可以自己的学号或其他名字建立文件夹；

(2) 启动 Visual C++ 环境，选择 File|New 菜单，弹出 New 对话框，在 Projects 页面选择 Win32 Console Application 工程类型，在 Project name 编辑框输入工程名 Ex1_1，路径选择自己的文件夹，如图 (1.3) 所示；

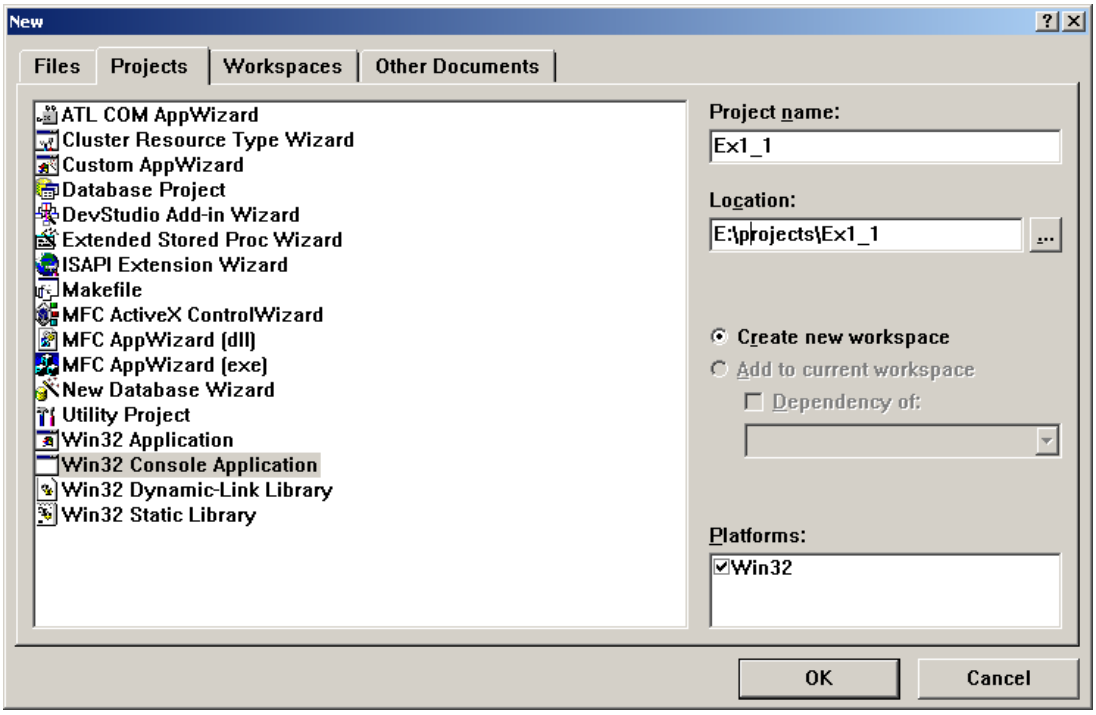


图 1.3: 创建新项目

(3) 按 OK 按钮，进入图 (1.4) 所示对话框，选择空项目；

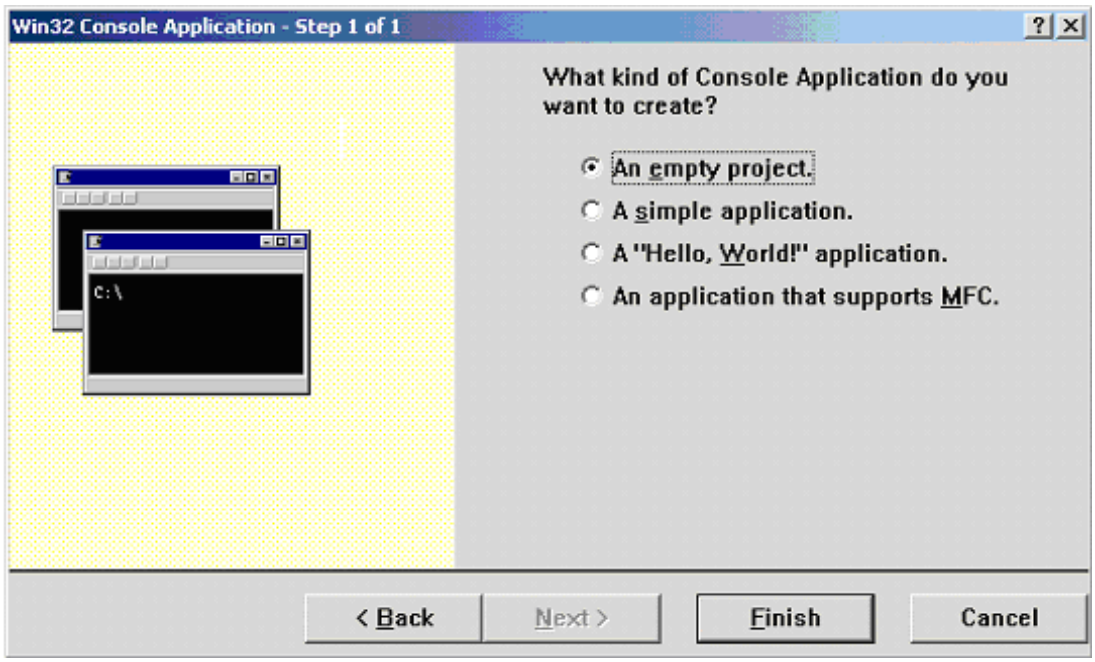


图 1.4: 选择空项目

(4) 按 Finish 按钮进入图 (1.5) 所示的 Project Information 框，它显示所创建项目的信息，确认后按 OK 就创建了一个名为 Ex1_1 的新项目，这是一个空的控

制台应用程序，且没有任何文件被添加到新工程中，此时，工程创建完成。

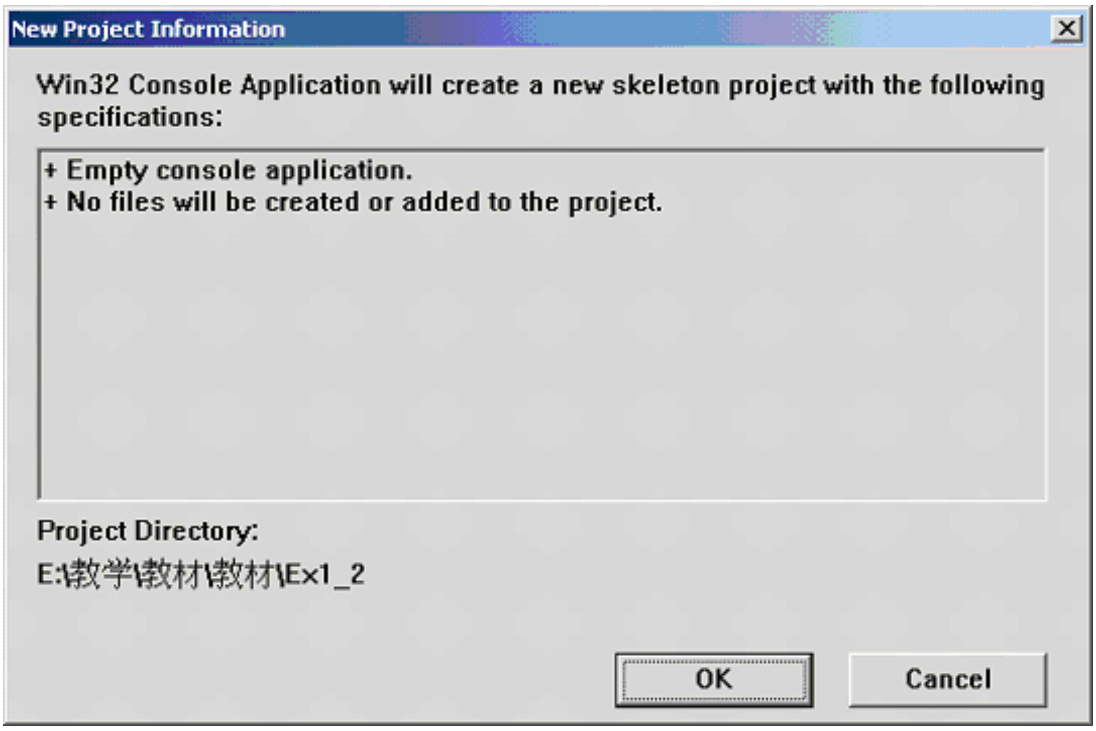


图 1.5: 新创建项目信息

**程序的编辑、编译、建立、执行**

选择 File|New 菜单项，在 New 对话框的 Files 页面选择 C++ Source File，输入文件名 Ex1_1.cpp，选中 Add to Project 复选框，如图 (1.6)；

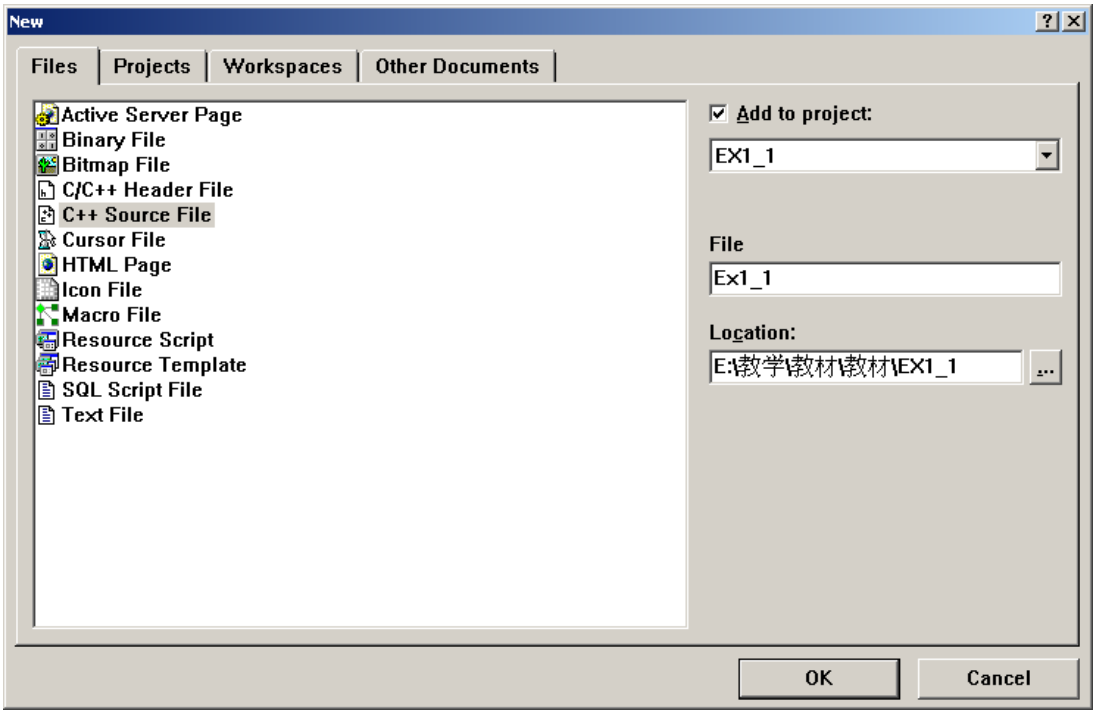


图 1.6: 创建新的 C++ 源文件

按 OK 按钮，打开了源文件编辑窗口，输入以下源代码：

```
#include <iostream.h>
void main(){
  int a,b,c;
  int max, min;
  cout<<"请输入三个不等整数:\n";
  cin>>a>>b>>c;
  if(a>b){ // A行
    max = a; min = b;
  } else { // B行
    max = b; min = a;
  }
  if(c>max)
    cout<<c<<'\t'<<max<<'\t'<<min<<'\n';
  else if(c<min)
    cout<<max<<'\t'<<min<<'\t'<<c<<'\n';
  else
    cout<<max<<'\t'<<c<<'\t'<<min<<'\n';
  return;
}
```

对于已经存在的源文件，选择 Project|Add to Project| Files…菜单项，在随后打开的插入文件对话框中选择待添加文件，按 OK 添加进工程；

**编译**

选择 Build|Compile 菜单项，即可编译源文件 Ex1_1.cpp，系统会在 Output 窗口

给出 Error(错误) 信息以及 Warning(警告) 信息。当所有 Error 改正后，得到目标文件 (Ex1_1.obj)。在编译过程中，编译器可能在 Output 窗口给出语法错误和编译错误信息：

(1) 语法错误 (Error) 处理：鼠标双击错误信息可跳转到错误源代码处进行修改，一个语法错误可能引发系统给出很多条 Error 信息，因此，发现一个错误并修改后最好重新编译一次，以便提高工作效率；

(2) 警告信息 (Warning) 处理：一般是触发了 C/C++ 的自动规则，如将一个浮点型数据给整型变量赋值，需要系统将浮点型数据自动转换为整型，此时小数部分会丢失，因而系统给出警告信息。警告信息不会影响程序执行，本例可以通过强制类型转换去掉警告信息。

**链接**

选择 Build|Build 菜单项，链接并建立工程的 exe 文件，得到可执行文件 Ex1_1.exe。在链接过程中，有时编译器可能会给出链接错误：

(1) 链接错误 (Linking Error) 处理：链接时可能产生错误，原因可能是所需要的库文件或目标文件缺少，或程序中调用的 Extern 函数没有定义等，只要补充相应文件再重新建立即可。

**执行**

选择 Build|Execute 菜单项，执行工程文件，会出现一个类似 DOS 操作系统的窗口，光标闪烁等待输入，按要求输入三个不等的整数后按 Enter 键，屏幕上由大到小输出这三个整数如图 (1.7)。


图 1.7: Ex1_1 运行结果

**程序调试**

运行程序，可能会发现程序没有编译错误，也能执行，但执行的结果不对，此时，除了仔细分析源程序，还可借助调试工具进行跟踪调试。

例如，在 Ex1_1 程序中 B 行处出错，在 else 后多加一个分号：

```
...
else; {max = b; min = a;}
...
```

用 88, 45, 67 这组数据测试，发现输出结果为 67 45 88，结果不对。下面介绍调试过程：

首先，在源程序中可能出现错误的行上设置断点，方法是将光标移至该行，然后按 F9 键，或选择工具栏上的手形按钮 (再按一次取消断点)，此时该行左侧出现一个红色圆点，断点设置成功，如图 8 在 A 行设置断点。

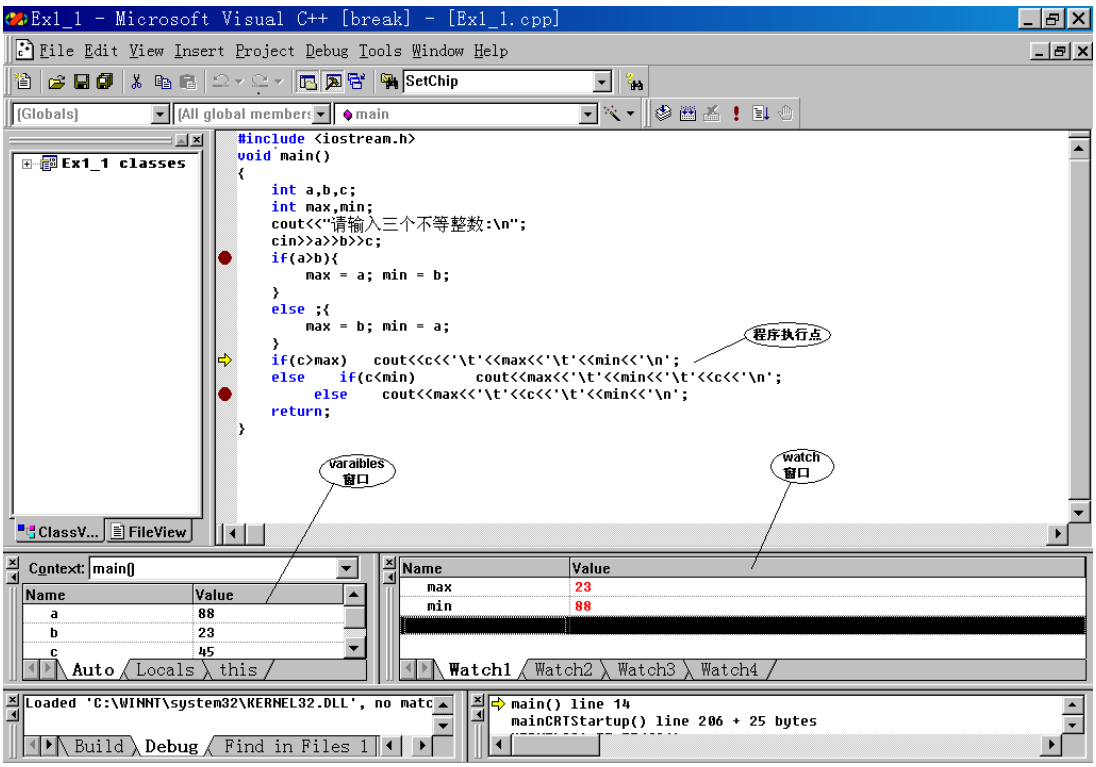


图 1.8: Visual C++　集成开发环境及程序的调试

选择 Build | Start Debug | Go…菜单命令 (也可选择 Build 工具栏上的 Go 图标)，程序执行到断点处停止，这时选择 View|Debug Windows 子菜单的 Watch 和 Variables 两个菜单项，打开监视和变量窗口观察变量值，分析查找出错原因。

在 Watch 窗口加入 max 和 min 两个变量，进行监视。Watch 窗口的每一行可显示一个变量，左栏显示变量名，双击它可进行编辑；右栏显示变量值。单步执行按 F10(不跟踪进函数) 或 F11(跟踪进函数内)，尽管 a>b, 在执行了 if 后面的 max = a; min = b; 后，仍然执行了 max = b; min = a;。当程序执行到箭头所指处时，max=23, min=88，如图 8，与预期结果不相符，说明程序的流程有问题。此时再仔细分析源程序，发现问题出在 else 后多余的分号；调试过程中 Variables 窗口动态显示各变量值随程序执行而

变化的结果。在学习了面向对象程序设计后，若程序中有类的对象，Variables 窗口的 this 页面可显示当前 this 指针所指向对象的各个值。修改源程序，再执行，反复调试，当程序中所有问题都得到改正后，得到正确的执行结果。

# 实验 2　函数

## 2.1　实验目的

(1) 理解函数概念，如何声明和定义函数；

(2) 理解按值传递参数和按引用传递参数的区别；

(3) 理解函数重载的概念，如何区分函数重载；

(4) 掌握引用初始化、引用作为函数参数；

(5) 掌握定义和使用函数指针；

## 2.2　实验要求

(1) 认真理解函数基本概念；

(2) 掌握按值传递参数和按引用传递参数及其区别；

(3) 在 Visual Studio6 环境下完成选做实验内容；

(4) 认真撰写完整的实验报告；

## 2.3　实验内容

(1) 定义一个函数，判断三个整型边长能否构成三角形，如果能判断它是否是直角三角形；

(2) 定义一个二维数组，数组维度由用户输入，通过函数求数组的鞍点 (二维数组的鞍点就是该位置上的元素在该行上是最大而在该列上是最小，数组可能没有鞍点)，函数参数分别为指针和引用的传址方式；

(3) 使用函数指针来求两个正整数的最大公约数和最小公倍数。

(4) 创建函数 plus()，它把两个数值加在一起，返回它们的和。提供 int、double、string 类型的重载版本，看能否处理下面的调用：

```
int n = plus(3, 4);
double d = plus(3.4, 4.7);
string str = plus(“aa”, “bbb”);
```

(5) 假定有如下形式的学生成绩表。编写函数 average 计算该成绩表的平均成绩和每个学生的平均成绩。在主函数输入该成绩表，调用该函数并输出结果。

| | 课程 1 | 课程 2 | 课程 3 | 课程 4 |
|---|---|---|---|---|
| 学生 1 | 77 | 68 | 86 | 73 |
| 学生 2 | 96 | 87 | 89 | 78 |
| 学生 3 | 70 | 90 | 86 | 81 |

表 2.1: 学生成绩

## 2.4 实验步骤

(1) 首先，声明和定义函数；

(2) 其次，对已经定义的函数进行重载，注意函数重载的要素；

(3) 然后进行函数参数传递实验，注意按值传递和按引用传递及其区别；

(4) 最后进行函数指针实验；

## 2.5 实验提示

(1) 注意给数组赋初始值；

(2) 注意数组越界问题；

# 实验 3 类和对象

## 3.1 实验目的

(1) 理解面向对象编程的思想，类和对象的基本概念；

(2) 理解类的成员的访问控制的含义，公有、私有和保护成员的区别；

(3) 掌握类对象的创建和初始化；

(4) 掌握构造函数和析构函数的含义与作用、定义方式和实现；

(5) 类的静态成员，静态成员变量和静态成员函数；

## 3.2 实验要求

(1) 复习类和对象的基本概念、C++ 中类的定义和对象创建过程；

(2) 使用 VC++ 的 debug 调试功能观察程序流程，跟踪观察类的构造函数、析构函数、成员函数的执行顺序；

(3) 在 Visual Studio6 环境下完成选做实验内容；

(4) 认真撰写完整的实验报告；

## 3.3 实验内容

(1) 定义一个圆类 (Circle)，属性为半径 (radius)，要求能获取和修改圆的半径，并能计算圆的周长和面积，并编写 main 函数进行测试。

(2) 设计一个程序，定义一个矩形类，包括数据成员和函数成员。要求能获取和修改矩阵的的长和宽，并能计算矩阵的面积，并编写 main 函数进行测试。

(3) 设计一个 Person 类，成员包括：姓名、性别、年龄。需要实现的功能 (成员函数)：输入、输出、修改成员，根据有关信息初始化对象。main() 函数先输出把对象初始化为缺省值的结果、再输出修改各成员的结果、再输出经输入函数修改各成员的结果、最后设计一个比较两人年龄的函数，并在 main() 函数中测试之。

(4) 定义一个 CPU 类，包含等级 (rank)、频率 (frequency)、电压 (voltage) 等属性，有两个公有成员函数 run、stop。其中，rank 为枚举类型 CPU_Rank，定义为：

```
enum CPU_Rank {P1=1, P2, P3, P4, P5, P6, P7};
```

frequency 为单位是 MHz 的整型数，voltage 为浮点型的电压值。观察构造函数和析构函数的调用顺序。

(5) 设计一个用于人事管理的 People(人员) 类。考虑到通用性，这里只抽象出所有类型人员都具有的属性：number(编号)、sex(性别)、birthday(出生日期)、id(身份证号) 等等。其中“出生日期”定义为一个“日期”类内嵌子对象。用成员函数实现对人员信息的录入和显示。要求包括构造函数和析构函数、拷贝构造函数、内联成员函数。

## 3.4 实验步骤

第四题实验步骤：

(1) 首先定义枚举类型 CPU_Rank，例如 enum CPU_Rank {P1=1, P2, P3, P4, P5, P6, P7}，再定义 CPU 类，包含等级 (rank)、频率 (frequency)、电压 (voltage) 等私有数据成员，定义成员函数 run、stop，用来输出提示信息，在构造函数和析构函数中也可以输出提示信息。在主程序中定义一个 CPU 的对象，调用其成员函数，观察类对象的构造与析构顺序，以及成员函数的调用。程序名：exp2_1.cpp。

(2) 使用 debug 调试功能观察程序 exp2_1.cpp 的运行流程，跟踪观察类的构造函数、析构函数、成员函数的执行顺序。参考程序如下：

```
// exp2_1.cpp
#include <iostream>
using namespace std;
enum CPU_Rank {P1=1,P2,P3,P4,P5,P6,P7};
class CPU
{
private:
  CPU_Rank rank;
  int frequency;
  float voltage;
public:
    CPU (CPU_Rank r, int f, float v)
  {
    rank = r;
    frequency = f;
    voltage = v;
    cout << "构造了一个CPU!" << endl;
  }
  ~CPU () { cout << "析构了一个CPU!" << endl; }

    CPU_Rank GetRank() const { return rank; }
    int GetFrequency() const { return frequency; }
   float GetVoltage() const { return voltage; }

    void SetRank(CPU_Rank r) { rank = r; }
    void SetFrequency(int f) { frequency = f; }
    void SetVoltage(float v) { voltage = v; }

    void Run() {cout << "CPU开始运行!" << endl; }
   void Stop() {cout << "CPU停止运行!" << endl; }
};

void main()
{
  CPU a(P6,300,2.8);
  a.Run();
  a.Stop();
```

```
}
```

(3) 调试操作步骤如下：

(3-1) 单击 Build | Start Debug | Step Into 命令，或按下快捷键 F11，系统进入单步执行状态，程序开始运行，一个 DOS 窗口出现，此时 Visual Studio 中光标停在 main() 函数的入口处；

(3-2) 从 Debug 菜单或 Debug 工具栏中单击 Step Over，此时，光标下移，程序准备执行 CPU 对象的初始化；

(3-3) 单击 Step Into，程序准备执行 CPU 类的构造函数；

(3-4) 连续单击 Step Over，观察构造函数的执行情况，直到执行完构造函数，程序回到主函数；

(3-5) 此时程序准备执行 CPU 对象的 run() 方法，单击 Step Into，程序进入 run() 成员函数，连续单击 Step Over，直到回到 main() 函数；

(3-6) 继续执行程序，参照上述的方法，观察程序的执行顺序，加深对类的构造函数、析构函数、成员函数的执行顺序的认识；

(3-7) 再试试 Debug 菜单栏中别的菜单项，熟悉 Debug 的各种方法。

其他题目实验步骤略。

## 3.5 实验提示

(1) 注意比较面向过程和面向对象编程；

(2) 理解类和对象；类是对同一类事物的抽象描述，其数据成员用于描述该类事物的属性，成员函数完成修改、获取属性值或实现基于属性的某些操作。类不占用存储空间。对象是类的实例，对象占用存储空间。C++ 中类与结构体并没有本质的区别，结构体中也可以定义成员函数，也可以指定各个成员的访问权限。两者的唯一差异在于：结构中成员的缺省访问权限是公有的，而类中成员的缺省访问权限是私有的。

(3) 类的定义与对象创建；

# 实验 4　运算符重载

## 4.1 实验目的

(1) 掌握友元函数的含义，友元函数和成员函数的区别；

(2) 掌握运算符重载为成员函数的方法；

(3) 掌握运算符重载为友元函数的方法；

## 4.2 实验要求

(1) 复习友元函数的定义，友元和成员函数之间有何区别；

(2) 复习运算符重载基本概念；

(3) 在 Visual Studio6 环境下完成实验内容；

(4) 认真撰写完整的实验报告；

## 4.3 实验内容

(1) 定义一个复数类 Complex，虚部和实部为私有数据类型。

(2) 重载加法运算符，减法运算符，+= 运算符。

(3) 重载一元负号和一元正号。

(4) 编写一个完整的程序，测试重载运算符的正确性。要求乘法“*”用友元函数实现重载，除法“/”用成员函数实现重载。

## 4.4 实验步骤

复数类声明参考实现如下：

```
Class complex{
Private:
  Double re, im;
Public:
  Complex(double r=0, double I=0): re(r),im(i){};
  Double real() const {return re;}
  Double imag() const {return im;}

  Complex& operator+=(complex);
  Complex& operator+=(double);
}

complex operator+(complex,complex);
complex operator+(complex,double);
complex operator+(double,complex);
complex operator- (complex,complex);
complex operator- (complex,double);
```

```
complex operator- (double,complex);

complex operator+(complex);
complex operator- (complex);

istream& operator>>(istream&, complex&);
ostream& operator>>(ostream&, complex&);
```

## 4.5 实验提示

(1) 注意比较使用两种不同的运算符重载方式；

# 实验 5 继承与派生

## 5.1 实验目的

(1) 理解继承的含义，掌握派生类的定义方法和实现；

(2) 掌握多重继承和派生类的方法；

(3) 掌握初始化基类成员的方法；

(4) 掌握定义虚基类的方法；

(5) 理解公有继承下基类成员对派生类成员和派生类对象的可访问性，能正确地访问继承层次中的各种类成员；

## 5.2 实验要求

(1) 认真复习继承和派生基本概念；

(2) 在 Visual Studio6 环境下完成选做实验内容；

(3) 认真撰写完整的实验报告；

## 5.3 实验内容

(1) 定义一个基类 Animal，有私有整型成员变量 age，构造其派生类 Dog，在其成员函数 setAge(int n) 中直接给 age 赋值，看看会有什么问题，把 age 改为公有成员变量，还会有问题吗？

(2) 编写一个程序计算出球、圆柱和圆锥的表面积和体积，要求如下：

(a) 定义一个基类圆，至少含有一个数据成员半径；

(b) 定义基类的派生类球、圆柱、圆锥，都含有求表面积和体积的成员函数和输出函数；

(c) 定义主函数，求球、圆柱、圆锥的体积；

(3) 定义一个日期 (年、月、日) 的类和一个时间 (时、分、秒) 的类，并由这两个类派生出日期和时间类。主函数完成基类和派生类的测试工作。

(4) 设计一个描述二维平面内一点的 CPoint 类，成员包括：x，y。需要实现的功能 (成员函数)：构造函数、设置坐标、获取坐标、获取极坐标、求两点之间的距离。然后再设计一个描叙三维平面内一点的 C3DPoint 类，该类为 CPoint 类的派生类，新添加的成员包括：z。需新添加或重新定义的功能包括：构造函数、设置坐标、获取坐标、求两点之间的距离。请在 main() 函数中测试之。

## 5.4 实验步骤

第一题步骤：

(1) 编写程序定义基类 Animal，成员变量 age 定义为私有的。构造派生类 Dog，在其成员函数 SetAge(int n) 中直接对 age 赋值时，会出现类似以下的错误提示：

```
error C2248: 'age' : cannot access private member declared in class 'Animal'
error C2248: 'age' : cannot access private member declared in class 'Animal'
```

(2) 把 age 改为公有成员变量后重新编译就可以了。

其他题目自己进行分析；

## 5.5 实验提示

(1) 注意派生类构造函数的定义；

(2) 写出程序，并调试程序，要给出测试数据和实验结果；

# 实验 6 虚函数与多态性

## 6.1 实验目的

(1) 理解虚函数在类的继承层次中的作用，虚函数的引入对程序运行时的影响

(2) 掌握使用虚函数实现动态多态性；

(3) 理解并掌握利用虚函数实现动态多态性和编写通用程序的方法；

## 6.2 实验要求

(1) 认真复习虚函数和多态性定义；

(2) 在 Visual Studio6 环境下完成选做实验内容；

(3) 认真撰写完整的实验报告；

## 6.3 实验内容

(1) 下面的 shape 类是一个表示形状的抽象类，area() 为求图形面积的函数。从 shape 类派生三角形类和圆类，并实现求面积函数：

```
class shape{
public:
  virtual double area()=0;
};
```

(2) 利用虚函数实现的多态性来求四种几何图形的面积之和。这四种几何图形是：三角形 (Triangle)、矩形 (Rectangle)、正方形 (Square) 和圆 (Circle)。几何图形的类型可以通过构造函数或通过成员函数来设置。

(3) 定义一个车 (Vehicle) 基类，有 run、stop 等成员函数，由此派生出自行车 (Bicycle) 类、汽车 (Motorcar) 类，从 Bicycle 和 Motorcar 派生出摩托车 (Motorcycle) 类，它们都有 run、stop 等成员函数。观察虚函数的作用。

## 6.4 实验步骤

第三题步骤：

(1) 编写程序定义一个车 (Vehicle) 基类，有 run、stop 等成员函数，由此派生出自行车 (Bicycle) 类、汽车 (Motorcar) 类，从 bicycle 和 motorcar 派生出摩托车 (Motorcycle) 类，它们都有 run、stop 等成员函数。

(2) 在 main() 函数中定义 Vehicle、Bicycle、Motorcar、Motorcycle 的对象，调用其 run()、stop() 函数，观察其执行情况。再分别用 Vehicle 类型的指针来调用这几个对象的成员函数，看看能否成功；

(3) 把 run、stop 定义为虚函数，再试试看；其他题目自己进行分析；

## 6.5 实验提示

(1) 虚函数和纯虚函数定义；

# 实验 7 异常处理

## 7.1 实验目的

(1) 理解异常的含义；

(2) 掌握异常处理的基本方法；

(3) 理解异常层次结构和使用 C++ 标准异常库的异常层次结构；

## 7.2 实验要求

(1) 理解异常基本概念；

(2) 在 Visual Studio6 环境下完成选做实验内容；

(3) 认真撰写完整的实验报告；

## 7.3 实验内容

(1) 为简单计算器程序添加异常处理，考虑被零除和输入非法字符等情况；

(2) 使用 C++ 标准异常类层次结构改写上题？

## 7.4 实验步骤

第一题

(1) 编写异常类，针对被零除和输入非法字符可以定义两种类型的异常；

(2) 定义构造函数，注意要输出异常类型；

第二题注意继承标准异常类；

## 7.5 实验提示

(1) 注意构造函数的定义；

# 实验 8　输入输出处理

## 8.1　实验目的

(1) 理解流和标准流的概念；

(2) 掌握创建和使用文件流；

(3) 掌握文件流的打开、关闭及使用的使用方法；

(4) 理解文本文件流与二进制文件流在操作上的区别；

## 8.2　实验要求

(1) 复习流的基本概念；

(2) 复习文本文件流与二进制文件流；

(3) 在 Visual Studio6 环境下完成选做实验内容；

(4) 认真撰写完整的实验报告；

## 8.3　实验内容

(1) 观察以下程序的输出，注意对输出格式的控制方法；

```
// exp6_1.cpp
#include <fstream>
using namespace std;
#define D(a) T << #a << endl; a
ofstream T("output.out");

void main() {
  D(int i = 53;)
  D(float f = 4700113.141593;)
  char* s = "Is there any more?";

  D(T.setf(ios::unitbuf);)

  D(T.setf(ios::showbase);)
  D(T.setf(ios::uppercase);)
  D(T.setf(ios::showpos);)
  D(T << i << endl;)
  D(T.setf(ios::hex, ios::basefield);)
  D(T << i << endl;)
  D(T.unsetf(ios::uppercase);)
  D(T.setf(ios::oct, ios::basefield);)
```

```
  D(T << i << endl;)
  D(T.unsetf(ios::showbase);)
  D(T.setf(ios::dec, ios::basefield);)
  D(T.setf(ios::left, ios::adjustfield);)
  D(T.fill('0');)
  D(T << "fill char: " << T.fill() << endl;)
  D(T.width(8);)
  T << i << endl;
  D(T.setf(ios::right, ios::adjustfield);)
  D(T.width(8);)
  T << i << endl;
  D(T.setf(ios::internal, ios::adjustfield);)
  D(T.width(8);)
  T << i << endl;
  D(T << i << endl;) // Without width(10)

  D(T.unsetf(ios::showpos);)
  D(T.setf(ios::showpoint);)
  D(T << "prec = " << T.precision() << endl;)
  D(T.setf(ios::scientific, ios::floatfield);)
  D(T << endl << f << endl;)
  D(T.setf(ios::fixed, ios::floatfield);)
  D(T << f << endl;)
  D(T.setf(0, ios::floatfield);) // Automatic
  D(T << f << endl;)
  D(T.precision(16);)
  D(T << "prec = " << T.precision() << endl;)
  D(T << endl << f << endl;)
  D(T.setf(ios::scientific, ios::floatfield);)
  D(T << endl << f << endl;)
  D(T.setf(ios::fixed, ios::floatfield);)
  D(T << f << endl;)
  D(T.setf(0, ios::floatfield);) // Automatic
  D(T << f << endl;)

  D(T.width(8);)
  T << s << endl;
  D(T.width(36);)
  T << s << endl;
  D(T.setf(ios::left, ios::adjustfield);)
  D(T.width(36);)
  T << s << endl;

  D(T.unsetf(ios::showpoint);)
  D(T.unsetf(ios::unitbuf);)
}
```

(2) 编写程序，用二进制方式打开指定的一个文件，在每一行前加行号。

(3) 产生 Fibonacci 数列的前 20 个数，使用文件流 ofstream 分别将它们写入文件 fibonacci.txt，再从文件中读入存放在数组中并输出到屏幕。

(4) 设计一个管理图书目的简单程序，提供的基本功能包括：可连续将新书存入文件“book.dat”中，新书信息加入到文件的尾部；也可以根据输入的书名进行查找；把文件“book.dat”中同书名的所有书显示出来。为简单起见，描述一本书的信息包括：书号，书名，出版社和作者。

## 8.4 实验步骤

第一题步骤：观察题目中程序的输出，学习对输出格式的控制方法；尝试更改输出语句中的参数，以加深对输出格式的理解；

第二题步骤：编写程序 exp6_2.cpp 使用 void main(int argc, char* argv[]) 函数中的参数传递操作的文件名，定义 ofstream 的对象对文件进行操作，使用 getline 成员函数读入数据，使用 cout 输出字符到文件；

## 8.5 实验提示

(1) 流的打开和关闭；

(2) 注意文件存储目录；

# 参考文献

[1] 刘建舟、徐承志等. *C++ 面向对象程序设计*. 机械工业出版社，http://www.hzbook.com, 2012.

[2] Author. *Title*. http://www.baidu.com, 2020.