



普通高等教育“十二五”计算机类规划教材

C++ 程序设计

基础教程

◎ 刘厚泉 李政伟 主编

*C++Chengxu Sheji
Jichu Jiaocheng*



机械工业出版社
CHINA MACHINE PRESS

普通高等教育“十二五”计算机类规划教材

C++ 程序设计基础教程

主 编 刘厚泉 李政伟

参 编 葛 欣 席景科 薛 猛 徐东红



机械工业出版社

全书共 13 章, 分别介绍了 C++ 基础、基本数据类型与运算符、程序控制、函数、数组、指针与引用、自定义数据类型、类、运算符重载、继承与派生、虚函数与多态性、C++ 输入/输出流。另外, 本书的附录提供了标准字符 ASCII 码表。本书内容精炼、讲解清晰、例题丰富、循序渐进。为便于教学, 每章都配有习题, 配套有《C++ 程序设计习题与上机指导》以及教学 PPT、例题源代码等, 帮助读者通过实践掌握 C++ 的编程方法。

本书是学习 C++ 的入门教程, 可以作为高等院校电子信息类专业程序设计基础的教材, 也适宜有兴趣学习 C++ 的非计算机专业学生作为辅助教材, 同时也适合自学。

图书在版编目 (CIP) 数据

C++ 程序设计基础教程/刘厚泉, 李政伟主编. —北京: 机械工业出版社, 2014. 5

普通高等教育“十二五”计算机类规划教材

ISBN 978-7-111-46506-5

I. ①C… II. ①刘… ②李… 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 (2014) 第 082751 号

机械工业出版社 (北京市百万庄大街 22 号 邮政编码 100037)

策划编辑: 刘丽敏 责任编辑: 刘丽敏 陈崇昱

版式设计: 赵颖喆 责任校对: 陈 越

封面设计: 张 静 责任印制: 李 洋

中国农业出版社印刷厂印刷

2014 年 7 月第 1 版第 1 次印刷

184 mm × 260 mm · 19.25 印张 · 471 千字

标准书号: ISBN 978-7-111-46506-5

定价: 38.00 元

凡购本书, 如有缺页、倒页、脱页, 由本社发行部调换

电话服务

网络服务

社服务中心: (010) 88361066 教材网: <http://www.cmpedu.com>

销售一部: (010) 68326294 机工官网: <http://www.cmpbook.com>

销售二部: (010) 88379649 机工官博: <http://weibo.com/cmp1952>

读者购书热线: (010) 88379203 封面无防伪标均为盗版

前 言

作为最流行的面向对象程序设计语言之一，C++ 既支持过程化程序设计，也支持面向对象程序设计。从操作系统、设备控制到数据库、网络、数字媒体等众多的系统软件和应用软件开发领域都能看到它的身影。

为了帮助读者更好地学习 C++ 程序设计，本书作者结合多年的 C++ 教学实践和软件开发经验，从零基础入手，全面、系统地介绍了 C++ 语言的相关知识，并辅以大量的程序实例。

全书共 13 章，分别介绍了 C++ 基础、基本数据类型与运算符、程序控制、函数、数组、指针与引用、自定义数据类型、类、运算符重载、继承与派生、虚函数与多态性、C++ 输入/输出流。本书具有如下特色：

(1) 详略得当，主次分明

C++ 中包含的内容较多，由于篇幅所限不可能面面俱到，必须有所取舍。书中对于非重点或较复杂的内容略讲，如数组部分重点是一维和二维数组，三维以上数组只介绍概念。另外，对一些选讲的内容以“*”标注。

(2) 讲解由浅入深，循序渐进

本书的编排采用循序渐进的方式，内容梯度从易到难，讲解由浅入深，适合各个层次的读者学习。

(3) 写作细致，以读者为出发点

本书内容编排、概念表述、语法讲解、实例展示，以及源代码注释等都很细致，力图将问题讲解得清楚，尽量为读者扫清学习中的障碍。

(4) 贯穿大量的典型实例和开发技巧

本书精选重点，强化主要概念，在讲解知识点时贯穿了大量的典型实例和开发技巧，以例题释含义，力求让读者获得真正实用的知识。

(5) 配套资源丰富

为方便教学和自学，本书配套有《C++ 程序设计习题与上机指导》，以及教学 PPT、例题源代码等，力求帮助读者通过具体实践掌握 C++ 的编程方法。

本书阅读建议：

(1) 建议没有基础的读者，从前向后顺次阅读，尽量不要跳跃。

(2) 书中的实例和习题要亲自上机动手实践，学习效果更好。

(3) 课后习题不仅要给出代码，最好能上机调试运行，以加强和巩固对知识点的理解和掌握。

本书由刘厚泉提出编写计划和结构安排，其中刘厚泉编写第 1~2 章，李政伟编写第 3~4、7 章，葛欣编写第 5~6 章、席景科编写第 10~11 章、薛猛编写第 8~9 章、徐东红编写第 12~13 章，最后由李政伟统稿。

本书的编写参考了大量的同类书籍。在此，我们向有关的作译者表示衷心的感谢。

由于 C++ 程序设计涉及的内容非常丰富，限于编者水平有限，书中个别地方难免有疏漏，敬请读者批评指正，如对本书有任何建议或意见，敬请来信 bookserviceofcpp@126.com。

编 者

目 录

前言

第1章 C++ 基础 1

- 1.1 程序设计语言简介 1
- 1.2 C++ 语言的特点与程序结构 2
- 1.3 过程型程序设计 6
- 1.4 面向对象程序设计 7
- 1.5 程序设计环境 9
- 1.6 小结 13
- 习题 14

第2章 基本数据类型与运算符 16

- 2.1 基本数据类型 16
- 2.2 常量 17
- 2.3 变量 21
- 2.4 赋值与算术运算符 23
- 2.5 逻辑运算符和关系运算符 25
- 2.6 其他运算符 28
- 2.7 运算符的优先级与结合性 28
- 2.8 数据类型的转换 29
- 2.9 小结 31
- 习题 31

第3章 程序控制 35

- 3.1 选择结构 35
- 3.2 循环结构 41
- 3.3 跳转结构 46
- 3.4 应用实例 48
- 3.5 小结 51
- 习题 51

第4章 函数 53

- 4.1 概述 53
- 4.2 函数的定义与声明 53
- 4.3 函数的调用 56
- 4.4 默认值参数 59
- 4.5 函数重载 60
- 4.6 函数模板 62

- 4.7 函数的嵌套调用 63

- 4.8 递归函数 66

- 4.9 作用域 68

- 4.10 存储类别 71

- 4.11 内部函数和外部函数 76

- 4.12 预处理指令 77

- 4.13 应用实例 79

- 4.14 小结 82

- 习题 82

第5章 数组 84

- 5.1 一维数组的定义与初始化 84

- 5.2 一维数组的使用 86

- 5.3 一维数组与函数 86

- 5.4 二维数组 89

- 5.5 字符数组 93

- 5.6 string 类型 98

- 5.7 应用实例 100

- 5.8 小结 106

- 习题 106

第6章 指针与引用 108

- 6.1 指针的定义与初始化 108

- 6.2 指针的使用 110

- 6.3 指针与数组 113

- 6.4 指针与字符串 119

- 6.5 指针与函数 121

- 6.6 引用 127

- 6.7 应用实例 130

- 6.8 小结 133

- 习题 134

第7章 自定义数据类型 135

- 7.1 结构体类型 135

- 7.2 结构体的使用 138

- 7.3 单向链表 143

- 7.4 共用体类型 150

- 7.5 枚举类型 151

7.6 类型定义 typedef	152	10.4 应用实例	222
7.7 小结	153	10.5 小结	228
习题	153	习题	229
第 8 章 类 (I)	155	第 11 章 继承与派生	231
8.1 类的定义	155	11.1 类的继承与派生	231
8.2 对象	162	11.2 类的继承方式	234
8.3 构造函数	165	11.3 派生类的构造函数和析构函数	242
8.4 析构函数	172	11.4 多重继承	245
8.5 静态成员	176	11.5 应用实例	252
8.6 应用实例	180	11.6 小结	256
8.7 小结	184	习题	257
习题	185	第 12 章 虚函数与多态性	260
第 9 章 类 (II)	188	12.1 多态性的概念	260
9.1 对象的存储	188	12.2 虚函数的定义	260
9.2 this 指针	189	12.3 纯虚函数与抽象类	264
9.3 信息的保护	191	12.4 应用实例	268
9.4 友元	197	12.5 小结	271
9.5 类模板	201	习题	271
9.6 * C++ 标准模板库 (STL)		第 13 章 C++ 输入/输出流	274
简介	204	13.1 C++ 的输入/输出流	274
9.7 应用实例	207	13.2 标准输入/输出流	275
9.8 小结	210	13.3 文件与文件流	286
习题	211	13.4 应用实例	295
第 10 章 运算符重载	212	13.5 小结	297
10.1 什么是运算符重载	212	习题	297
10.2 运算符重载规则	214	附录 标准字符 ASCII 码表	300
10.3 运算符重载的实现形式	215	参考文献	301

第1章 C++ 基础

作为一种广泛使用的面向对象的程序设计语言，C++ 语言继承了 C 语言全部的优点和功能，并支持面向对象的程序设计风格。C++ 既是一种十分流行和实用的高级语言，也是学习面向对象程序设计的首选语言。本章主要介绍程序设计语言的发展概况、C++ 语言的基本特点，以及 C++ 语言程序的开发过程。

1.1 程序设计语言简介

1.1.1 程序设计语言是人机交互的重要介质

人们在使用计算机解决复杂问题时，需要通过程序表达意图和步骤，这通常包含两个阶段，第一个阶段是对问题进行分析和描述，明确做什么和怎么做；第二个阶段是使用程序设计语言编写程序，将解决问题的方法和步骤交由计算机执行。学好程序设计也包含两方面的内容，一方面要熟练掌握程序设计语言的使用方法，另一方面也要学会用计算机思维来分析和表达现实世界的问题。

1.1.2 程序设计语言的发展概况

计算机程序设计语言的发展，经历了从机器语言、汇编语言到高级语言的历程。

(1) 机器语言

机器语言是第一代计算机语言。机器语言程序由一串串使用“0”和“1”编码的指令序列构成，计算机硬件可以直接理解这种二进制形式表示的指令，并完成相应的操作。机器语言依赖于特定型号的计算机硬件，具有很高的运行效率。虽然对机器来说理解和执行这种语言都十分简单，但是对程序员来说程序的编写、阅读和修改都很困难，而且由于每台计算机的指令系统各不相同，在一台计算机上执行的程序很难移植到另一台计算机上执行，因此加重了程序设计者的负担。

(2) 汇编语言

汇编语言是第二代计算机语言。为了减轻使用机器语言编程的不便，人们尝试用一些简洁的英文符号来替代指令中特定的二进制串，比如用“ADD”表示加法，用“MOV”表示数据传递等。这种用英文和数字符号组成的语言称为汇编语言。与机器语言相比，汇编语言程序易于阅读和理解，方便了程序的编写和维护。计算机并不能直接理解汇编语言的指令，在程序运行之前，需要通过专门的程序将其翻译成二进制形式的机器语言程序，再交由机器执行。汇编语言也同样十分依赖于机器硬件，虽然移植性不好，但是程序精炼而质量高，所以至今仍在使用。

(3) 高级语言

由于使用机器语言和汇编语言开发程序效率低且易出错，人们开始寻找一种接近于数学



或自然语言的编程语言，这种语言不依赖于计算机硬件，编出的程序能在不同的机器上通用。经过努力，第一个完全脱离机器硬件的高级语言 FORTRAN 于 1954 年问世。从此以后，共有几百种高级语言出现，影响较大且使用较普遍使用的有 FORTRAN、ALGOL、COBOL、BASIC、LISP、Pascal、C/C++、C#、PROLOG、Ada、Java 等。高级语言的发展也经历了从早期的无结构语言到结构化语言，从过程型语言到面向对象语言的历程。相应的，软件的开发也由最初的个体手工作坊式的封闭式生产，发展为产业化、流水线式的工业化生产。

1.2 C++ 语言的特点与程序结构

1.2.1 C++ 语言的特点

C++ 语言由 C 语言发展而来，它保留了 C 语言高效、灵活、便于移植等特点，又增加了对面向对象程序设计的支持。C++ 语言适合于各种应用软件和系统软件的程序设计，用它编写的程序结构清晰、易于扩充、代码质量高。C 和 C++ 的主要特点是：

(1) 语句简练、程序结构简单。C++ 语言书写格式自由，语法结构清晰、紧凑。出于语言的简洁和运行效率等方面的考虑，C++ 的绝大部分功能都由各种标准函数和类来实现。

(2) 数据类型丰富、齐全。标准 C 提供了整数、实数、字符、字符串等基本数据类型，还提供数组、指针、结构体等构造数据类型。C++ 还增加了类这一特殊数据类型的定义机制。

(3) 运算符丰富、齐全，运算能力强，具有直接的硬件处理能力。C/C++ 提供的运算符分为常规运算和与硬件有关的运算两部分。常规运算符包括算术运算符、逻辑运算符、关系运算符等，这类运算符各种语言一般都具备；与硬件有关的运算符包括位运算符、地址运算符等，这些运算符是 C/C++ 所独有的，体现出汇编语言的某些特征。

(4) 语言的通用性及可移植性强，程序执行效率高。

1.2.2 简单的 C++ 程序

为了使读者了解 C++ 程序的特点，下面先介绍几个简单的程序。

【例 1-1】 输出一串字符 “I Love C++”。

```
#include <iostream>           //编译预处理指令
using namespace std;          //使用命名空间 std
int main()
{
    cout << "I Love C++";      //输出字符串
    return 0;
}
```

运行结果如下：

```
I Love C++
```

程序的第 1 行 “#include <iostream>” 是一个“文件包含指令”。它是 C++ 的编译预处理

理指令之一，其作用是将文件 `iostream` 的内容包含到该指令所在的程序文件中，代替该指令行。该指令行以“#”开头，行末尾没有分号，以区别于普通的C++语句。

程序的第2行“`using namespace std;`”的意思是使用命名空间 `std`。C++ 标准库中的类和函数在命名空间 `std` 中进行了声明。

`main` 是 C/C++ 的主函数的名字。每一个C++ 程序都必须有且只有一个 `main` 函数。`main` 前面的 `int` 声明了函数的类型为整型。

由花括号 `{}` 括起来的部分为函数体。本例中主函数内包含一个以 `cout` 开头的语句。`cout` 是C++ 系统定义的对象，称为输出流对象。为便于理解和使用，把用 `cout` 和“`<<`”完成输出的语句简称为 `cout` 语句。注意C++ 所有语句最后都有一个分号“`;`”。

`return` 语句的作用是向操作系统返回一个零值，表明程序正常结束。

【例 1-2】 输出两个整数之和与差。

```
#include <iostream>           //编译预处理指令
using namespace std;          //使用命名空间 std
int main()                     //主函数头部
{                               //函数体开始
    int a,b;                   //定义变量 a 和 b
    int x,y;                   //定义变量 x 和 y
    cin >> a >> b;             //输入 a 和 b 的值
    x = a + b;                 //计算 a + b 并赋给 x
    y = a - b;                 //计算 a - b 并赋给 y
    cout << "x = " << x << endl; //输出 x 的值
    cout << "y = " << y << endl; //输出 y 的值
    return 0;                 //程序正常结束时向操作系统返回零值
}                               //函数体结束
```

本程序计算两个整数 `a` 和 `b` 之和与差。程序中出现的以“`//`”开始的部分为注释，表示从它开始到本行末尾之间的全部内容都将作为注释。注释是源程序的一部分，以便能更好地理解程序，但在程序编译时系统会忽略所有注释，因此对程序运行不起作用。

程序中定义的变量 `a`、`b` 和 `x`、`y` 均为整型 (`int`) 变量。`cin` 是C++ 系统定义的输入流对象，“`>>`”是“提取运算符”，与 `cin` 配合使用，其作用是从输入设备中（如键盘）提取数据送到输入流 `cin` 中。把用 `cin` 和“`>>`”实现输入的语句简称为 `cin` 语句。在执行 `cin` 语句时，从键盘输入的第1个数据被赋给整型变量 `a`，输入的第2个数据被赋给整型变量 `b`。两个数据之间需要用空格或回车分割。第8行和第9行将 `a + b` 的值赋给整型变量 `x`，将 `a - b` 的值赋给整型变量 `y`。第10行和第11行分别输出 `x` 和 `y` 的值，`cout` 语句中的 `endl` 是C++ 输出时的控制符，作用是换行。

如果输入：

123 111 ✓

运行结果为

x = 234
y = 12

**【例 1-3】**按从大到小的顺序输出两个整数。

```

#include <iostream>                                //预处理指令
using namespace std;
int print(int x,int y)                             //定义 print 函数，函数值为整型，形式参数 x, y
                                                    //为整型
{
    int z;                                         //局部变量声明
    if(x<y)                                       //判断如果 x 小于 y
    {                                             //则交换 x 和 y 的值
        z=x;
        x=y;
        y=z;
    }
    cout << x << ' ' << y << endl;              //按从大到小顺序输出两个数的值
    return 0;                                    //返回语句
}                                                  //print 函数结束

int main()                                         //主函数
{
    int a,b,m;                                    //定义变量
    cin >> a >> b;                                //输入变量 a 和 b 的值
    print(a,b);                                   //调用 print 函数,a 和 b 作为函数的实参数
    return 0;                                     //程序正常结束时向操作系统返回一个零值
}

```

本程序包括两个函数：主函数 main 和被调用的函数 print。

输入数据如下。（注意输入的两个数据之间可用空格或回车间隔）

50 80 ↵

运行结果为

80 50

【例 1-4】面向对象程序设计示例。

```

#include <iostream>
#include <cmath>
using namespace std;
class Line                                         //声明一个线段类,类名为 Line
{
private:                                         //类中的私有成员
    int x1,y1;                                  //私有变量 x1, y1
    int x2,y2;                                  //私有变量 x2, y2
public:                                          //类中的公有成员
    void input ()                               //定义公有函数 input
    {
        cin >> x1 >> y1;                       //输入 x1, y1 的值
        cin >> x2 >> y2;                       //输入 x2, y2 的值
    }
}

```

```

void getLen()                                //定义公有函数 getLen
{
    double len;
    len = sqrt((x1 - x2) * (x1 - x2) + (y1 - y2) * (y1 - y2));
    cout << "length = " << len << endl;    //输出线段长度
}

};                                           //类的声明结束
int main()                                  //主函数首部
{
    Line line;                               //定义 line 为 Line 类的对象
    line.input();                             //调用对象 line 的 input 函数
    line.getLen();                           //调用对象 line 的 getLen 函数
    return 0;
}

```

一个类中一般包含两种成员：数据和函数，分别称为数据成员和成员函数。或者说，类由若干数据以及对其操作的函数组成。类体现了数据的封装性和信息隐蔽。在上面的程序中，在声明 Line 类时，把类中的数据和函数分为两大类：private（私有的）和 public（公用的）。把全部数据（x1, y1, x2, y2）指定为私有的，把全部函数（input, getLen）指定为公有的。具有“类”类型特征的变量称为“对象”（object）。

输入数据（输入线段的起点和终点）：

```
10 10 13 14 ✓
```

运行结果如下：

```
length = 5
```

1.2.3 C++ 程序的结构

C++ 程序的结构和书写格式归纳如下：

(1) C++ 程序可以由一个或多个编译单元构成，每一个编译单元为一个文件。在程序编译时，编译系统分别对各个文件进行编译。

(2) 一个文件中可以包括预处理指令、全局声明（在函数外的声明部分）、函数等部分。函数是程序最基本的组成部分。每一个程序必须包括一个或多个函数，其中必须有一个（而且只能有一个）主函数（main 函数）。

(3) 一个函数由函数头部和函数体组成。函数头部即函数的第一行，包括函数名、函数类型、参数（包括类型及参数名）。函数参数可以默认，如 int main（）。函数体即函数头部下面的花括号及其内部的部分，一般包括局部声明部分（在函数内的声明部分）和执行语句部分。

(4) C++ 语句以分号结束。语句包括两类，一类是声明语句，另一类是执行语句。C++ 对每一种语句赋予一种特定的功能。语句是实现操作的基本成分，没有语句的函数是没有意义的。

(5) 一个 C++ 程序总是从 main 函数开始执行，而不论 main 函数在整个程序中的位置如何。

(6) 类（class）是 C++ 新增加的重要的数据类型，是 C++ 对 C 的最重要的发展。有了



类，就可以实现面向对象程序设计方法中的信息隐蔽、继承、多态等功能。在一个类中可以包括数据成员和成员函数，它们可以被指定为私有的（private）和公用的（public）。私有的数据成员和成员函数只能被本类的成员函数所使用，公用的数据成员和成员函数则可以被其他类或函数使用。

(7) 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。

(8) 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。用“//”作注释时，有效范围只有一行，即本行有效，不能跨行。而用“/*……*/”作注释时有效范围为多行。只要在开始处有一个“/*”，在最后一行结束处有一个“*/”即可。一般来说，内容较少的单行注释常用“//”，内容较长的多行注释则常用“/*……*/”。

1.3 过程型程序设计

1.3.1 过程型程序设计的特点

一个过程型程序应包括以下两方面内容：

- 对数据的描述。在程序中要指定数据的类型和数据的组织形式，即数据结构（Data Structure）。

- 对操作的描述。即操作步骤，也就是算法（Algorithm）。

即可以用下面的公式表示：程序 = 算法 + 数据结构。

过程型程序设计采用模块分解与功能抽象方法。对复杂的程序设计任务采取自顶向下分解，按功能划分为若干个基本模块，形成一个树状结构，如图 1-1 所示。各模块间的关系尽可能简单，功能上相对独立。每一模块内部均是由顺序、选择和循环三种基本结构组成。

过程型程序设计的优点是能够有效地将一个较复杂的任务分解成许多易于控制和处理的子任务，便于开发和维护。相对于面向对象程序设计而言，过程型程序设计的缺点是可重用性差、数据安全性差、不便于开发大型软件和以图形界面为主的应用软件。

1.3.2 程序设计流程图

编写程序解决问题往往需要从两个方面考虑：一是要明确问题中隐含了哪些数据；二是针对这些数据，采用什么方法完成计算。因此一个程序也应该包括两方面的内容：一方面对数据进行描述，指定数据的类型和数据之间的关系（即数据结构）；另一方面对数据的操作进行描述，根据操作步骤书写出能实现操作的语句序列（算法）。

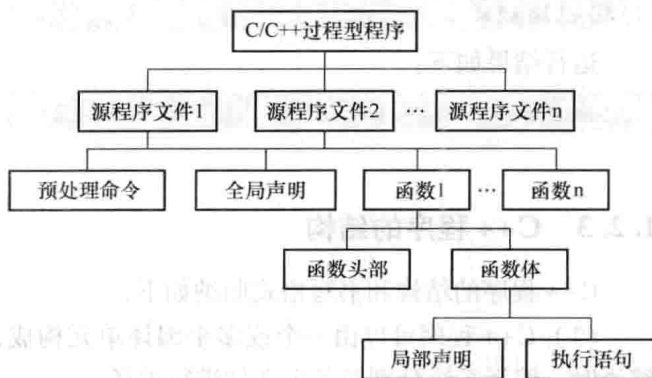


图 1-1 过程型程序的结构

算法描述是一个由整体到局部、由粗到精的过程。算法的描述方法有：自然语言表示法、流程图表示法、N-S 盒图表示法、伪代码表示法、计算机语言表示法等。

自然语言就是人们日常使用的语言，可以是汉语、英语或其他语言。用自然语言表示算法，则算法通俗易懂，但文字冗长，容易出现歧义。自然语言所描述的内容往往不太严格，要根据上下文才能判断其正确含义。在算法分析的初始阶段使用自然语言描述算法是恰当的，尤其对于初学者来说，这是学习程序设计时所迈出的第一步。

流程图使用一些图形符号来表示算法中的各种操作。与自然语言相比，使用流程图来描述算法更加简洁直观，易于理解。流程图中常用的图形符号如图 1-2 所示。在图 1-2 中菱形框的作用是对一个给定的条件进行逻辑判断，根据给定的条件是否成立来决定如何执行其后的操作。连接点（小圆圈）用于将画在不同地方的流程线连接起来。使用连接点，可以避免流程线交叉或者过长，使流程图更加清晰。注释框不是流程图的必要部分，不反映流程和操作，只是为了对流程图中某些操作进行必要的补充说明，以帮助人们阅读和理解流程图。

例如，求整数 1 至 1000 的累加和。用变量 i 分步存储 1 ~ 1000 的数值，用 s 存储每一步累加之后的和， i 的初始值为 1， s 的初始值为 0。将 i 的数值累加到 s 中后，判断 i 的值是否大于 1000，如果大于 1000，则停止计算，否则重复相同操作。相应算法描述如图 1-3 所示。

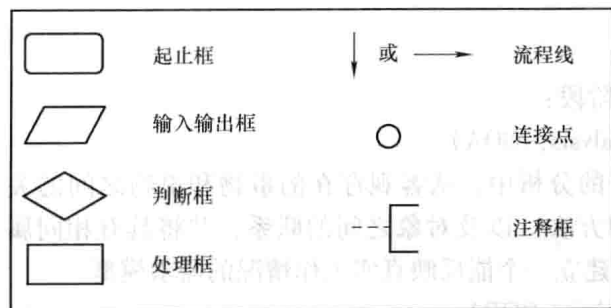


图 1-2 流程图常用的图形符号

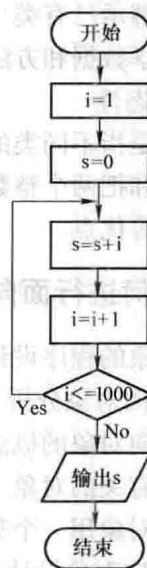


图 1-3 求整数 1~1000 的累加和

1.4 面向对象程序设计

面向对象的程序设计方法最基本的思想就是把客观世界看成一个一个相对独立而又相互联系的实体（称为对象）。例如，一个桌子、一个气球都是一个对象。类是对象集合的抽象，规定了这些对象的公共属性和方法，对象是类的一个实例。例如，钢笔是一个类，具体到一支特定的钢笔就是一个对象。C++ 支持面向对象的程序设计方法，适合于大、中型软件的开发，它从开发时间、费用到软件的可重用性、可扩充性、可维护性和可靠性等方面，都具有很大的优越性。



1.4.1 面向对象程序设计的特点

面向对象程序设计具有抽象、封装、继承和多态性 4 种基本特征。

(1) 抽象

抽象就是忽略一个主题中与当前目标无关的那些方面，以便充分地注意与当前目标有关的方面。“对象”就是一个提取实际事物中与问题相关的属性和方法，并能在程序设计中直接使用的抽象事物。

(2) 封装

封装是把数据（属性）和函数（方法）封藏起来，当成一个有机整体。封装是面向对象的重要特征，它首先实现了数据隐藏，保护了对象的数据不被外界随意改变；其次它使得对象成了相对独立的功能模块。

(3) 继承

自然界中的大部分事物之间都有很多共性，但也有不同。比如，四边形是一个类，而矩形与四边形有相同的性质，也有自己的属性。将事物之间的共性保留下来就是继承。如矩形继承四边形这个类的公共属性，但同时又进一步定义了新的特性。面向对象程序设计方法允许一个新类继承已有类（称为基类）的属性和方法，该新类称为派生类。继承是类的层次结构之间共享数据和方法的机制，实现了类的重用。

(4) 多态性

多态性是指不同类的对象对同一消息作出不同的响应。例如，同样是加法，把两个字符串加在一起和把两个整数加在一起的内涵是完全不同的。多态性具有灵活、抽象、行为共享和代码共享等优点。

1.4.2 如何进行面向对象程序设计

面向对象的程序设计过程包括以下几个阶段：

(1) 面向对象分析（Object Oriented Analysis, OOA）

按照面向对象的概念和方法，在对任务的分析中，从客观存在的事物和事物之间的关系，归纳出有关的对象（包括对象的属性和方法）以及对象之间的联系，并将具有相同属性和行为的对象用一个类（class）来表示。建立一个能反映真实工作情况的需求模型。

(2) 面向对象设计（Object Oriented Design, OOD）

根据面向对象分析阶段形成的需求模型，对每一部分分别进行具体的设计。首先是进行类的设计，类的设计可能包含多个层次（利用继承与派生）。然后以这些类为基础得出程序设计的思路和方法，包括对算法的设计。

(3) 面向对象编程（Object Oriented Programming, OOP）

根据面向对象设计的结果，用一种计算机语言把它写成程序。C++ 是面向对象编程的理想语言。

(4) 面向对象测试（Object Oriented Test, OOT）

在写好程序并交给用户使用前，必须对程序进行严格地测试。测试的目的是发现程序中的错误并改正它。面向对象测试是用面向对象的方法进行测试，以类作为测试的基本单元。

(5) 面向对象维护 (Object Oriented Soft Maintenance, OOSM)

由于对象的封装性, 因此修改一个对象对其他对象的影响很小。利用面向对象的方法来维护程序, 大大提高了软件维护的效率。

1.5 程序设计环境

1.5.1 程序集成开发环境的功能

程序集成开发环境可以把各种编程需要的功能有机地结合起来, 统一在一个图形化操作界面下, 为程序设计人员提供尽可能高效、便利的服务。在程序设计过程中为了排除错误, 需要反复进行编译—查错—修改—再编译的循环, 集成开发环境可以使各步骤之间方便快捷地切换。输入源程序后用菜单命令或快捷键启动编译, 出现错误后又能立即转到对源程序进行修改, 甚至直接把光标定位到出错的位置上。编辑器除了具备一般文本编辑器的基本功能外, 还能根据 C++ 的语法规则, 自动识别程序文本中不同的语法成分, 并且用不同的颜色显示不同的成分, 对使用者产生很好的提示效果。下面主要简介 Visual C++ 6.0 (以下简称 VC 6) 开发环境的使用方法。

1.5.2 程序集成开发环境的使用

(1) 启动并进入 Visual C++ 6.0 的开发环境。

通过“开始”→“程序”→“Microsoft Visual C++ 6.0”, 进入到 Microsoft Visual C++ 6.0 集成开发环境窗口, 其窗口样式如图 1-4 所示。

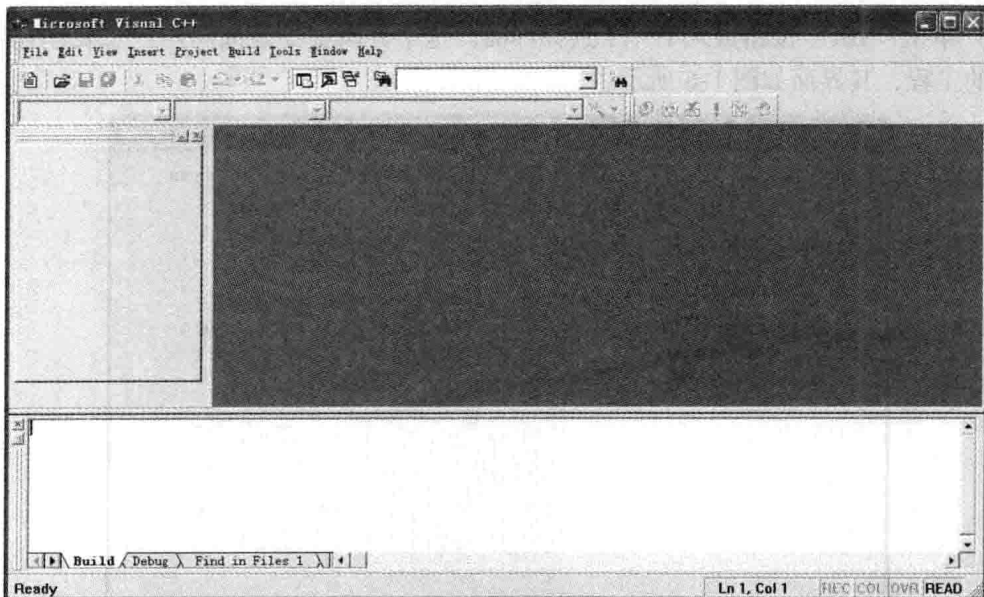


图 1-4 启动并进入 Visual C++ 6.0 开发环境

窗口大体上可分为 4 部分。上部: 菜单和工具栏; 中左: 工作区 (Workspace) 显示窗口, 这里将显示处理过程中与项目相关的各种文件等信息; 中右: 视图区, 是显示和编辑程

序文件的操作区；下部：输出（Output）窗口区，程序调试过程中，进行编译、链接、运行时输出的相关信息将在此处显示。

（2）选择“File”菜单下的“New”项，会出现一个选择界面，在属性页中选择“Projects”标签后，会看到近 20 种的工程类型，只需选择其中最简单的一种：“Win32 Console Application”，然后往右上方的“Location”文本框和“Project name”文本框中填入工程存放的磁盘位置（目录或文件夹位置）以及工程名称，界面如图 1-5 所示。

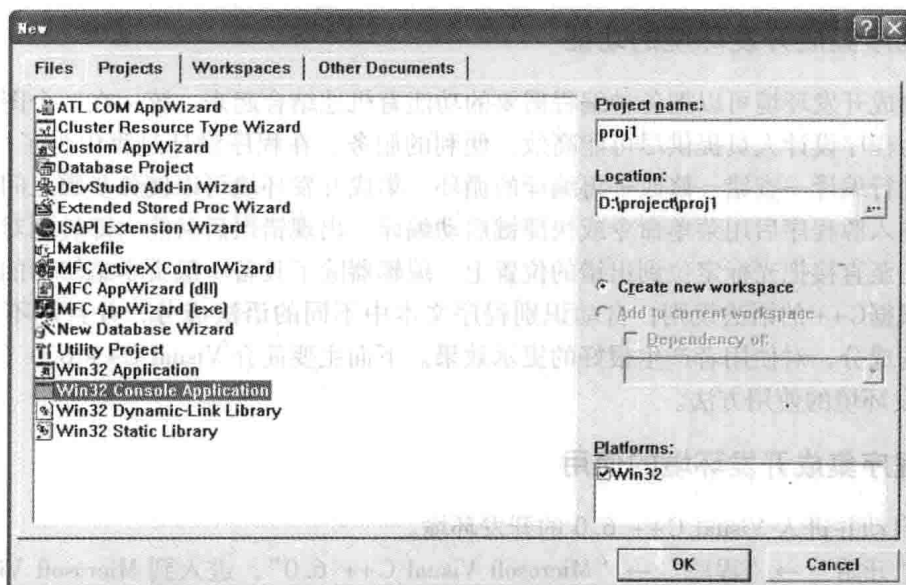


图 1-5 建立工程

（3）单击“OK”按钮进入下一个选择界面。这个界面主要是询问用户想要创建一个什么类型的工程，其界面如图 1-6 所示。

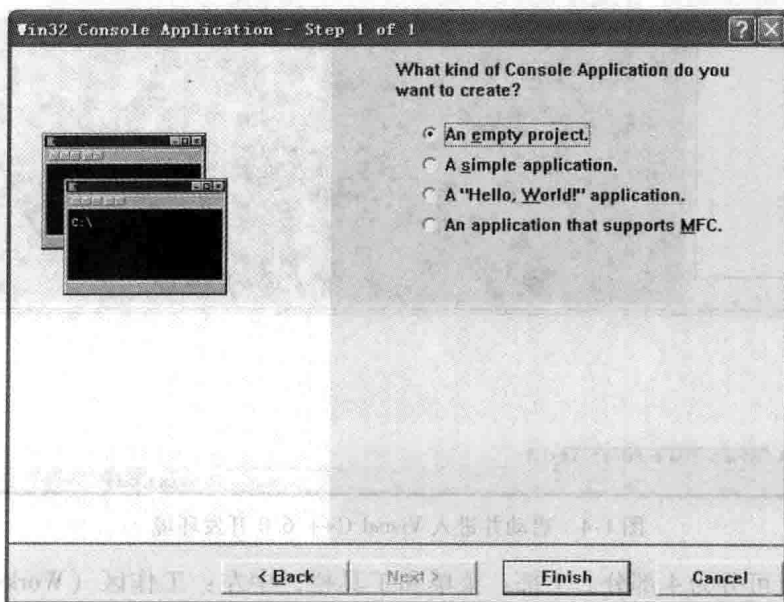


图 1-6 工程类型选择

若选择 “An empty project” 项将生成一个空的工程，工程内不包括任何东西；

若选择 “A simple application” 项将生成包含一个空的 main 函数和一个空的头文件的工程；

若选择 “A ‘Hello World!’ application” 项与选择 “A simple application” 项没有什么本质的区别，这个工程只是包含有显示出 “Hello World!” 字符串的输出语句；

若选择 “An application that supports MFC” 项，可以利用 VC6 所提供的类库来进行编程。

这里选择 “An empty project” 项，以便从一个空的工程开始我们的工作。

(4) 单击 “Finish” 按钮，就进入真正的编程环境下了，界面如图 1-7 所示。

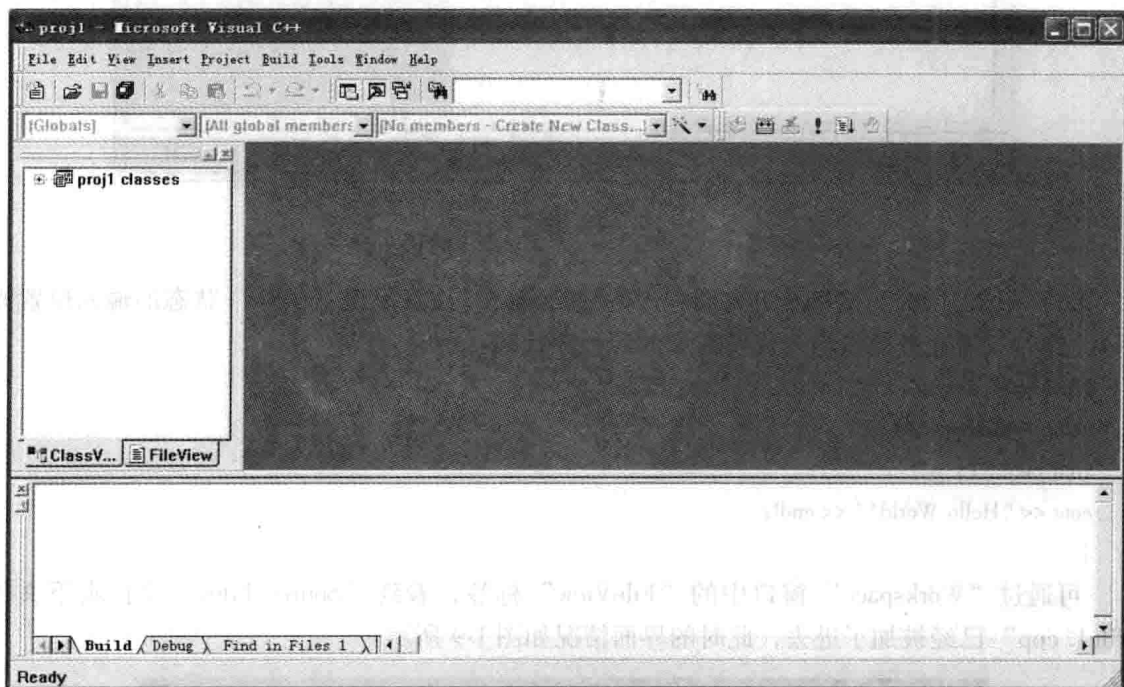


图 1-7 工程已建立

注意屏幕左中的 “Workspace” 窗口，该窗口中有两个标签，一个是 “ClassView”，一个是 “FileView”。“ClassView” 中列出的是这个工程中所包含的所有类的有关信息，“FileView” 中列出的是这个工程所包含的所有文件信息。单击 “+” 图标打开对应的层次会发现 3 个文件夹：“Source Files” 文件夹中包含了工程中所有的源文件；“Header Files” 文件夹中包含了工程中所有的头文件；“Resource Files” 文件夹中包含了工程中所有的资源文件。

(5) 选择 “Project” 菜单，依次执行 “Add To Project” → “New” 命令，在属性页中选择 “Files” 标签，然后选择 “C++ Source File” 项，在右中处的 “File” 文本框中为将要生成的文件取一个名字，我们取名为 “file1”（其他遵照系统默认设置），此时的界面情况如图 1-8 所示。