高等院校信息技术规划教材

C++程序设计

张军　编著

清华大学出版社

高等院校信息技术规划教材

C++程序设计

张军　编著

清华大学出版社
北京

内 容 简 介

本书在全面介绍 C++ 语言中面向过程方法的语法知识的基础上，着重介绍面向对象程序设计方法中的类、对象、继承、派生和多态性的概念。本书以程序的运行时间和存储空间为主线，把握程序的静态性和动态性两个特点，运用“运行时序图”和“内存模型图”的分析手段，从时间和空间两个角度深入讲解程序运行的基本原理，使读者掌握面向对象程序设计的思想和方法。

全书共分 3 部分：第 1 部分（第 1、2 章）为 C++ 的基础知识，着重介绍 C++ 语言的基本知识；第 2 部分（第 3～6 章）为 C++ 面向过程的程序设计，着重介绍 C++ 语言中的操作符、语句、函数、数组、指针和引用；第 3 部分（第 7～9 章）为 C++ 面向对象的程序设计，着重介绍 C++ 语言中的类、对象、继承与派生和多态性。全书提供了大量应用实例，每章后均附有习题。

所有实例代码都在 VC++ 6.0 编译器运行通过。为了便于教学，作者还为本书制作了配套的电子课件。

本书适合作为高等学校计算机程序设计课程的教学用书，同时可供 C++ 程序设计开发人员和研究人员参考。

本书封面贴有清华大学出版社防伪标签，无标签者不得销售。
版权所有，侵权必究。侵权举报电话：010-62782989　13701121933

图书在版编目（CIP）数据

C++ 程序设计/张军编著. --北京：清华大学出版社，2016
高等院校信息技术规划教材
ISBN 978-7-302-43635-5

Ⅰ. ①C…　Ⅱ. ①张…　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（2016）第 083535 号

责任编辑：焦　虹
封面设计：常雪影
责任校对：焦丽丽
责任印制：何　芊

出版发行：清华大学出版社
网　　址：http://www.tup.com.cn，http://www.wqbook.com
地　　址：北京清华大学学研大厦 A 座　　**邮　　编**：100084
社 总 机：010-62770175　　**邮　　购**：010-62786544
投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn
质量反馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
课件下载：http://www.tup.com.cn，010-62795954
印 装 者：北京鑫海金澳胶印有限公司
经　　销：全国新华书店
开　　本：185mm×260mm　　**印　　张**：20.5　　**字　　数**：474 千字
版　　次：2016 年 7 月第 1 版　　**印　　次**：2016 年 7 月第 1 次印刷
印　　数：1～2000
定　　价：39.80 元

产品编号：063553-01

前言 foreword

全国大多数高校都把C语言或者C++语言作为学习程序设计的入门课程，一般都是在大学第一学期或者第二学期就开始进入程序设计语言的学习。在学习程序设计语言之前，大多数高校仅仅开设了“计算机导论”课程，因此初学者对计算机专业知识了解很少，在学习程序设计语言时，只能片面地、肤浅地用学习数学的方法来学习编程语言。例如，数学中的等式：

$$a+b=c$$

在C++程序中表现为下列语句：

```
a+b=c;
```

但在编译时就会出错：“'=' : left operand must be l-value”。意思是说：等号左边必须是一个值。其实这里的值就是指变量。为什么数学上可以这样写，但在C++程序中就不行呢？

显然，在C++中有等号的式子，编译器都会当成赋值语句来处理。这时，大多数读者可能就会死记这条语法规则：在赋值语句中，等号左边必须是一个变量。为什么要求是变量？也许大多数初学者只知其然，不知其所以然。其实，对于赋值语句计算机是这样处理的：先计算等号右边式子的值，再将值存放到等号左边表示的内存单元中。在计算机中，只有变量才会分配内存单元用于存放数据值，因此等号左边必须是一个变量。前面的语句中由于等号左边的式子a+b没有内存单元，所以编译时就会出错。程序中，只有定义了变量，操作系统才会为这个变量分配内存单元，变量才能存储数据。这也是为什么在C++程序中变量必须“先定义后使用”的原因。

又如，在学习数组时，对数组元素的访问有两种方法：指针法和下标法。例如：

```
int a[10];
```

下面两种语句是等价的：

```
a[5]=1;
*(a+5)=1;
```

这两种对数组元素的赋值方法，哪种执行效率高呢？

将数组元素存储在一块连续的存储空间里就可以随机地访问它们。在大多数程序设计语言中，一个具有 n 个元素的数组中的元素是按照 0,1,…,n－1 编号的。假设每个数组元素的宽度是 w，那么数组 a 的第 i 个元素的开始地址为

$$base+i\times w$$

其中 base 是分配给数组 a 的内存块的开始地址。也就是说，base 是 a[0]的开始地址，因此

$$a[5]的地址=a[0]的地址+5\times 4$$

这个计算过程需要 CPU 参与运算。

第二种赋值语句中，*(a＋5)表示的内存地址只需要 CPU 中的指令指针寄存器 IP 向下移动 5 个内存块(每个内存块的宽度为一个数组元素的宽度)，这个过程不需要 CPU 参与运算。

从以上分析可以得知，第二种赋值语句，即指针法的执行效率较高。要理解其中的道理，就必须了解计算机组成原理和编译原理的相关知识。

从上面两个例子可以知道，程序设计语言涉及专业知识非常广泛，只单一地学习计算机程序设计语言的语法知识是很难真正学懂计算机程序设计语言的。这些专业知识正是初学者所缺乏的，也正是初学者学习程序设计语言感觉非常困难的原因之一。

由于上述原因，初学者对程序设计语言的学习往往还停留在死记硬背语法规则，对程序设计语言的理解还停留在“只知其一，不知其二”的基础上。由于初学者不能很好地掌握程序设计语言的原理，对语言的“迁移”能力很差，因此大多数高校开设了多门程序设计语言课程，浪费了大量课时，这也是当前大多数程序设计课程的基本现状。因此，在学习程序设计语言之前，有必要了解程序的相关知识。

1. 程序＝数据结构＋算法

程序是一组指令的有效集合。数据是被程序处理的信息。由于数据在计算机中通过程序指令来处理，因此就要规定数据的组织形式——数据结构。所谓数据结构，是指相互之间存在一种或多种特定关系的数据元素的集合，它是计算机存储、组织数据的方式。每一种数据结构都有存储结构和逻辑结构。存储结构就是指数据在内存中的存储方式，逻辑结构就是指数据的组织形式。算法(algorithm)是对解题方案的准确、完整的描述，是一系列解决问题的清晰指令。通俗地说，算法就是计算方法。算法和数据结构的结合构成了程序。因此，程序也可以理解为是用一定的数据结构对算法的实现，通常用某种程序设计语言编写，运行于某种目标体系结构上。打个比方，一个程序就像一个用汉语(程序设计语言)写下菜谱(程序)，用于指导懂汉语和烹饪方法的人(体系结构)来做这个菜。

2. 程序≠进程

程序是一组有序的静态指令，是一种静态的概念。所谓进程，就是程序的一次运行活动，属于一种动态的概念。进程离开了程序也就没有了存在的意义。因此，可以这样说：程序是进程运行的静态文本，而进程是执行程序的动态过程。如果把一部动画片的电影拷贝比拟成一个程序，那么这部动画片的一次放映过程就可比作一个进程。

试读结束：需要全本请在线购买： www.ertongbook.com

一个进程可以执行一个或多个程序。例如，一个编译进程进行C++源程序编译时，要执行词法分析、语法分析、语义分析、代码优化、存储分配和代码生成等几个程序。反之，同一程序也可能由多个进程同时执行。例如，上述C++编译程序可能同时被几个编译进程执行，它们对相同或不同的源程序分别进行编译，各自产生目标程序。再次以动画片及其放映活动为例，一次电影放映活动可以连续放映几部动画片，相当于一个进程可以执行几个程序。反之，一部动画片可以同时在若干家电影院中放映，这相当于多个进程可以执行同一程序。不过要注意的是，几家电影院放映同一部电影，如果使用的是同一份拷贝，那么实际上是交叉进行的。但在多处理机情况下，几个进程却完全可以同时使用一个程序副本。

程序可以作为一种软件资源长期保存，而进程则是一次执行过程，它是暂时的，是动态地产生和终止的。这相当于电影拷贝可以长期保存，而一次放映活动却只延续1～2h。

进程在内存中有其相应的代码空间(代码区)和数据空间(数据区)，而程序存储在机器的外部存储器中，不会占用内存空间。进程需要使用一种机构才能执行程序，这种机构称为处理机(processor)。处理机执行指令，根据指令的性质，处理机可以单独用硬件或软、硬件结合起来构成。如果指令是机器指令，那么处理机就是一般所说的中央处理机(CPU)。

3. 逻辑地址≠物理地址

任何程序在没有得到运行前，都以静态文本的形式存放在计算机的外部存储器中。这些静态文本只有逻辑地址。程序从外存调入内存运行时，要给程序分配一定的内存空间。这些内存空间都有物理地址，也就是说，程序具有了物理地址。这时，程序就变成了进程，从静态文本变成了动态程序。为了保证CPU执行指令时可以正确地访问存储单元，需将用户程序中的逻辑地址转换为运行时由机器直接寻址的物理地址，这一过程称为地址映射或者地址绑定。一个程序在运行时刻，逻辑地址空间的映像包括数据区和代码区，如第2章的图2.12所示。对地址的进一步了解可以参见2.2.2节、2.2.3节、2.2.6节、7.4.5节和9.1节。

生命在于运动，程序也是一样。只有运行中的程序才具有生命力，程序中的各种变量才会拥有内存空间，它们才会有生命周期。程序要得到运行，就得有现实的物质条件：运行时间和内存空间。程序只有分配了物理内存空间，得到了CPU的运行时间，才能真正得以运行。算法是程序的灵魂，对算法的评价主要从时间复杂度和空间复杂度来考虑。同样，对程序的认识也应当从运行时间和内存空间两个角度去考虑。因为一个程序只有得到时间和空间两个物质条件才会得到执行，才会变得有意义。

本书以程序的运行时间和存储空间(时空)为主线，把握程序的静态性和动态性两个特点，通过“运行时序图”和“内存模型图”的分析手段，从时间和空间两个角度深入讲解程序运行时的基本原理。在讲解程序设计语言基本概念的同时，穿插讲述计算机组成原理、操作系统、编译原理、数据结构、算法设计等方面的相关知识。

C语言是面向过程的程序设计语言，C++是从C语言发展演变而来的一种面向对象的程序设计语言。那么，学习C++语言是否应该首先学习C语言呢？答案是否定的。不应该把面向过程和面向对象的程序设计对立起来，任何面向对象的程序设计都需要用到

面向过程的知识。其实，C++并不是纯粹的面向对象的语言，它是一种混合语言。C++既可以编写面向过程的程序，也可以编写面向对象的程序。因此，C++语言也可以作为程序设计入门语言来学习。

本书共分3部分：

第1部分(第1、2章)：C++的基础知识。首先通过简单例子介绍一个C++程序的结构特点：程序由若干个函数组成；然后介绍面向过程与面向对象这两种程序设计方法的区别及特点；最后介绍C++语言的字符集、标识符、关键字和数据类型。目的是让读者对C++程序设计有一个总体认识。

第2部分(第3～6章)：C++面向过程的程序设计。首先介绍操作符的优先级和结合性；然后介绍3种语句——选择、循环、跳转，数组、指针和引用的使用方法；最后介绍函数。

第3部分(第7～9章)：C++面向对象的程序设计。主要介绍面向对象程序设计的4个特点：抽象、封装、继承、多态性。

本书的大多数示例都是笔者在多年教学过程中的演示程序，简单易懂。由于不同学校、不同专业对学习C++程序设计有不同的要求，因此对于学过C语言的学生，教师在使用本教材时，对第1、2部分可以只用少量课时对符号常量、引用、string类、内联函数、函数重载和函数模板等知识进行讲解，重点应放在第3部分的学习中。

本书肯定会有不妥甚至错误之处，诚请广大读者、教师不吝指正，作者将不胜感激。作者的电子邮件地址：teacherzj@126.com。

张　军

2016年5月

目录 contents

试读结束：需要全本请在线购买： www.ertongbook.com

第1章 chapter 1

认识C++

1.1 C++ 简单程序设计

不论程序的规模只有几行或是成千上万行，C++ 程序都是由几个基本部分组成的。现在从最简单的“hello world!”程序开始。

1.1.1 “hello world!”程序实例

例 1.1 一个简单的 C++ 程序。

```
//*************************************
//The First C++program
//*************************************
#include<iostream.h>
int main()
{
    cout<<"Hello world!";
    return 0;
}
```

1. 注释

第 1、2、3 行：注释行(comments)。注释通常用于概述算法，说明变量的用途，或者解释晦涩难懂的代码。编译器会忽略注释，因此注释对程序的行为或性能不会有任何影响。

在 C++ 中有两种注释：单行注释和多行注释。

单行注释以双斜线(//)开始，以换行符结束。当前行双斜线右侧的所有内容都会被编译器忽略，这种注释可以包含任何文本，包括额外的双斜线。

多行注释使用继承自 C 语言的两个限定符(/* 和 */)。这种注释以/* 开始，以 */结束，可以包含除 */外的任何内容，包括换行符。编译器将在/* 和 */之间的所有内容都当作注释。

2. 预处理指令

第4行：#include<iostream.h>为编译预处理指令(preprocessor)，指示编译器对程序进行处理(预处理指令将在7.10节中详细介绍)。将文件iostream.h中的代码嵌入到程序中该指令所在的地方。文件iostream.h中声明了程序所需要的输入和输出操作的有关信息。cout和"<<"操作的有关信息就是在该文件中声明的。

3. main函数

第5行：int main()为主函数的头部(head)，也可称为函数原型。

每个C++程序都包含一个或多个函数(function)，其中一个必须命名为main，即主函数。操作系统通过调用main函数来运行C++程序。

一个函数的定义包括四个部分：返回类型(return type)、函数名(function name)、一对括号包围的参数列表(parameter list，允许为空)和一对大括号(curly brace)包围的函数体(function body)。返回类型、函数名和参数列表通常又称为函数原型。函数调用完毕后，要用return语句返回一个与返回类型相匹配的值。

函数定义的语法形式为：

```
Datatype FunctionName(Datatype1 variable1,Datatype2 variable2,…)
{
    statement
    …
}
```

其中：

Datatype：函数返回类型，只能有一种返回类型。

FunctionName：函数名。

Datatype i(i=1,2,…)：函数参数类型，可以有多个参数。

variable：参数名。

Datatype FunctionName(Datatype1 variable1,Datatype2 variable2,…)：函数原型。

在本例中，main函数的返回类型为int，即整数类型。

4. 函数体

第6～9行：主函数的函数体。它是一个以左花括号开始，以右花括号结束的语句块(block of statement)。

```
{
    cout<<"hello world!";
    return 0;
}
```

5. C++语句

语句(statement)是C++程序的执行部分,每个语句由分号(;)作为结束符。每种程序设计语言都包含了各种丰富的语句,以实现复杂的程序功能。这里的函数体中只包含了两个语句。

第7行:cout输出语句。cout是一个输出流对象(对象的概念在7.2节中介绍),它是C++系统预定义的对象,其中包含了许多输出功能。输出操作由操作符"<<"来表达,其作用是将紧随其后的双引号中的字符串输出到标准输出设备(显示器)上。

第8行:return语句。当return语句包括一个返回值时,此值的类型必须与函数的返回类型相匹配。在本例中,main的返回类型为int,而返回值为0也是一个int类型的值。大多数系统中,main的返回值通常被用来指示状态。当程序正常运行结束时,向操作系统返回的值为0。非0的返回值的含义由系统定义,通常用来指示错误类型。

例1.1运行时在屏幕上输出如下结果:

```
Hello world!
```

1.1.2 综合程序实例

例1.2 一个综合实例。

```
//*************************************
//The second C++program
//*************************************
#include<iostream>
using namespace std;                    //使用命名空间 std
void Print()                            //定义一个函数 Print,返回值为空,没有参数
{
    cout<<"hello world!"<<endl;         //输出"hello world!",然后换行
    cout<<"welcome to c++"<<endl;       //输出 welcome to c++,然后换行
}
void main()                             //主函数头部
{
    Print();                            //函数调用语句
}
```

程序运行结果:

```
hello world!
welcome to c++
```

第5行:使用命名空间std,C++标准库中的类和函数是在命名空间std中声明的,因此程序中如果需要用到C++标准库,就要用这条指令来声明。有关命名空间的概念将在2.2.5节中详细介绍。

第6行:定义一个函数,函数名为Print,返回类型为空,参数列表为空。所以在函数

体中不用 return 语句来返回值。

第 7～10 行：函数体。当执行

```
cout<<" hello world!"<<endl;
```

时，首先在屏幕中输出字符串“hello world!”。endl 为 end of line 的缩写，表示换行符，因此光标转到下一行。当执行：

```
cout<<" welcome to c++"<<endl ";
```

时，在屏幕中输出：welcome to c++ ，然后换行。

第 13 行：函数调用语句。一个函数要得到执行，必须通过主函数或其他函数进行调用。调用一个函数前，必须先定义。

函数调用的语法形式为

```
FunctionName(Expression1, Expression2,…);
```

其中：

Expression：函数实际参数。

(Expression1,Expression2,…)：实参列表(ArgumentList)。

这里的实参列表必须和函数定义时参数列表的个数和类型相匹配。这里的 Print 函数由于定义时没有参数列表，所以调用时也就没有参数。函数的进一步使用在第 6 章中再详细介绍。

1.2 C 和 C++

对于计算机语言初学者来说，有很多关于 C 和 C++ 的误解。C 和 C++ 是两种不同的语言吗？C 和 C++ 有什么区别和联系？可以直接学习 C++ 吗？

C 语言最早由贝尔实验室开发并由美国国家标准化协会 ANSI(American National Standard Institute) 于 1989 年进行了标准化。1990 年，国际标准化组织 ISO (International Standard Organization)采纳了 ANSI 标准。C 语言是一种典型的面向过程(procedure-oriented,OP)的程序设计语言。后来，人们对 C 语言进行了扩充，使其能够被用来开发面向对象(object-oriented,OO)的软件。学习 C++ 语言，同时也是在学习 C 语言。C++ 语言完全兼容 C 语言，继承了 C 语言的一切语法。比如：if 语句和 for 语句在 C 和 C++ 中基本相同；C 有结构(structure)，C++ 有结构(structure)和类(class)；C++ 有函数重载、函数模板和操作符重载，而 C 没有。结构、函数、类，现在读者不需要知道这些概念，不过，知道 C++ 包含 C 的全部功能，同时还提供了更加丰富的功能，这一点很重要。

1.2.1 面向过程与面向对象

1. 面向过程

过程就是思考或解决问题的步骤，在程序中就是指函数。面向过程就是一种以过程

试读结束：需要全本请在线购买： www.ertongbook.com