初耀军　编著

C++程序设计基础及实践

21世纪计算机科学与技术实践型教程

丛书主编　陈明

清华大学出版社

初耀军　编著

C++程序设计基础及实践

清华大学出版社
北京

21世纪计算机科学与技术实践型教程
丛书主编　陈明

内 容 简 介

本书实际是一本基础知识和项目相结合的教科书，但是为了符合一般读者的学习和思维习惯，所以采用传统章节组织方式和内容安排。它从C++编程规范出发，引入设计模式和项目设计。遵循读者认知规律，以循序渐进、由浅入深的讲解方式，使读者在C++基础知识、基本方法、基本技能、项目设计、编程规范等方面奠定一定基础。

全书共分13章，各章均配有练习、项目设计，项目的全部代码放在资料中，旨在突出主题，知识系统。

本书可作高等院校的C++编程教材，尤其可作为项目式教学的教材，也可作二级C++考试、自学考试参考资料。

图书在版编目(CIP)数据

C++程序设计基础及实践/初耀军编著. —北京：清华大学出版社，2016
21世纪计算机科学与技术实践型教程
ISBN 978-7-302-43560-0

Ⅰ. ①C…　Ⅱ. ①初…　Ⅲ. ①C语言－程序设计　Ⅳ. ①TP312

中国版本图书馆CIP数据核字(2016)第081956号

责任编辑：谢　琛
封面设计：常雪影
责任校对：梁　毅
责任印制：宋　林

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,010-62770175-4608
印 刷 者：北京富博印刷有限公司
装 订 者：北京市密云县京文制本装订厂
经　　销：全国新华书店
开　　本：185mm×260mm　　**印　　张**：25　　**字　　数**：595千字
版　　次：2016年10月第1版　　**印　　次**：2016年10月第1次印刷
印　　数：1～2000
定　　价：49.00元

产品编号：064501-01

前　言

本书将 C++ 基础知识与项目开发相结合，以培养读者的计算思维。

实践由两部分构成。基础知识部分，主要由 C++ 考试真题构成。项目设计中，主要是模仿教材中的项目，设计自己的项目，使学生了解企业文化。

本书的编写，主要考虑以下几个方面。

1. 读者对象

本书主要面对高校各专业的大学生，尤其是高职院校学生，兼顾参加全国计算机二级 C++ 考试和全国自学考试的读者。

读者学习 C++ 的根本目的是初步掌握面向对象编程方法，能够利用 C++ 编写一定程度的程序，为以后进一步学习和应用 C++ 打下良好的基础，同时初步熟悉项目开发的基本思路和方法。

内容选取侧重初学者的实际，主要选取 C++ 二级考试和全国自学考试内容，使读者学会面向对象 C++ 编程的基本内容、基本方法和基本技能。

2. 内容安排

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。

以面向过程的编程作为切入点，从编写简单的程序开始，由面向过程到面向对象，循序渐进、逐步深入，比较符合读者的认识规律，每一步的台阶都比较小，学习难度不大，读者容易理解。同时也按“自顶向下，逐步求精”的原则进行项目设计。

一个实际的 C++ 项目设计需要考虑许多因素，但是为教学需要，将项目代码作为每章附录，这样避免冲淡教学内容，使知识体系完整，符合一般读者的学习习惯。因此，在教学内容主体上，对项目作了简化。有些在专业人员看来很“幼稚”的程序，但在学习者看来可能是一个很合适的教学程序。在初步掌握 C++ 编程方法后，可以逐步使程序复杂些，长一些，更接近真实程序一些。在学完本课程后，最好完成一个实际的应用程序，以提高实际应用的能力。

以注册登录的设计为背景，使学生了解企业文化、描述工具 UML。在程序风格和质量上参照《华为 C++ 编码规范》，力求编写高质量程序，使读者养成良好的编程习惯。

3. 教材体系

本书全面而系统地介绍 C++ 的主要功能，引导读者由简而繁地学会编写 C++ 程序。有了 C++ 编程的初步基础后，再进一步提高，掌握更多更深入的算法。这样的编写方法

可能符合大多数学习者的情况，降低了学习难度。

同时考虑项目设计过程，将知识体系与项目设计过程融合，使读者在掌握 C++ 基础知识的同时了解项目开发思路和技巧。

4. 项目选取

本书毕竟是教授 C++ 编程，以语言本身为重点，为使读者学以致用，引进项目。之所以选择注册登录，主要因为项目设计还涉及《软件工程》、《项目管理》和《软件测试》等有关方面内容，如果完全按项目设计过程安排，内容会更加庞大，主次不分。一般读者比较熟悉注册和登录，减少系统分析时间，优化项目以符合教学要求。

5. 巩固知识

编程课是一门实践性很强的课程，只靠听课和看书是不够的。衡量学习效果好坏的标准不是“懂不懂”，而是“会不会设计”。因此必须强调多编程，多上机实践。

本书得到了许多同志和朋友的帮助和支持。在构思上，高云、殷晓春、周乃富老师提出了许多宝贵意见；在代码设计上，殷晓春、闫冰、董志勇老师给予大力协助；在实践环节上，周乃富、韩金华老师给出许多建议，还有出版社的老师也给予宝贵建议和大力支持，在此表示衷心的感谢！

由于作者水平有限，加之时间仓促，疏漏在所难免，恳请广大读者不吝赐教，以便在今后的版本中进行改进。

作　者

2015 年 12 月

目　录

第1章 C++语言基础

教学目标：

(1) 了解C++的面向对象特征。

(2) 了解源程序和头文件的基本结构、main函数的作用。

(3) 掌握C++程序的基本框架和主函数的语法格式。

(4) 掌握C++语言的词汇：关键字、标识符、常量、运算符和标点符号等。

(5) 掌握注释及其应用。

(6) 应用源程序的编程规范。

1.1 面向对象的特征

1.1.1 什么是面向对象

在编程中，OOP是英文Object Oriented Programming的缩写，即面向对象编程或面向对象程序设计。面向对象编程是一种编程方法，它以对象和类为基础。

对象(Object)是现实世界中客观存在的事物，一个对象就是一个描述客观事物的实体，如一个人是“人”中的一个对象、一本书是“书”中一个对象。

类(class)是对一组性质相同对象的抽象描述，如对人、书的描述等。类是用户定义的一种数据类型，是面向对象编程的核心。“物以类聚，人以群分”，描述了面向对象中的对象和类。

面向对象有三个主要特征：**封装性、继承性和多态性**。

(1) 封装性。封装性是一种隐藏对象的属性和实现细节的隐藏技术。在C++中，所谓封装就是将一组数据及其相关操作(算法)捆绑成一个整体，该整体就是对象。描述对象的数据及其相关操作被封装在其内部。C++通过“类”来封装、隐藏信息。

(2) 继承性。继承性是指一种事物保留了另一种事物的全部或部分特征，并且具有自身的独有特征。继承是将另一个类的属性和行为全部或部分接受过来，为适应发展又添加了自己特有的属性和行为。

继承创建的新类，称为“子类”或“派生类”，本书统一称为“子类”；被继承的类称为“基类”、“父类”或“超类”，本书统一称为“基类”。

继承的过程是从一般到特殊的过程。通过“继承(泛化)”(Inheritance)和“组合(聚合)”(Composition)来实现继承。

(3) 多态性。多态性是指当多种事物继承于一种事物时,同一种操作在它们之间表现出不同的行为。不同的对象调用相同名称的方法,并可导致完全不同的行为的现象称多态性。通过覆盖和重载实现多态。

封装可隐藏实现的细节,使代码模块化。继承可扩展已存在的代码模块(类),它们的目的都是为了代码重用。多态是为了实现接口的重用。

1.1.2 C++ 面向对象的特征

C++ 是一种面向对象的编程语言,C++ 支持的面向对象特征。

(1) 支持数据封装,亦即支持数据抽象。在 C++ 中,类是支持数据封装的工具,对象则是数据封装的实现。在面向对象的编程中,将数据和对该数据进行合法操作的函数封装在一起,作为一个类的定义,数据将被隐藏在封装体中,该封装体通过操作接口与外界交换信息。对象被说明具有一个给定类的“变量”。C++ 中的类是数据和函数的封装体,结构体类型可作为一种特殊的类。

(2) 类中包含私有、公有和保护成员。私有成员,只有在类中说明的函数才能访问该类的私有成员,而在该类外的函数不可访问私有成员;公有成员,本类内、本类的子类及该类外面函数都可访问公有成员,称为该类的接口;保护成员,只有该类及该类的子类中的函数可访问,其余的在这个类外的函数均不能访问。

(3) 通过发送消息来处理对象。C++ 中每个对象根据所接收到的消息的性质来决定需要采取什么样的行动,以响应这个消息。响应这个消息由一系列的方法完成,方法是在类定义中用函数来定义的,使用一种类似于“函数调用”的机制把消息发送到一个对象上。

(4) 允许友元(友元函数,友元类)破坏封装在类中的私有成员。一般,私有成员不允许该类外面的任何函数访问,但友元可访问该类的私有成员(包含数据成员和成员函数)。友元可以是在类外定义的普通函数,也可以是在类外定义的一个类,前者称友元函数,后者称为友元类。友元打破了类的封装性,它是 C++ 另一个面向对象的重要性。

(5) 支持多态性。C++ 允许一个相同的标识符或运算符代表多个不同实现的函数,称标识符或运算符的重载,用户可以根据需要定义标识符重载或运算符重载。通过定义函数重载来支持静态联编。

(6) 支持继承性。C++ 中允许单继承和多继承。一个类可根据需要生成派生类。派生类自身还可定义所需要的不包含在父类中的新成员函数。一个子类的每个对象包含有从父类那里继承来的数据成员以及自己所特有的数据成员。

(7) 支持动态联编。C++ 中可定义虚函数,通过定义虚函数来支持动态联编。

以上是 C++ 对面向对象编程中的一些主要特征的支持。

1.2　C++ 应用程序的组成

通常，在一个 C++ 程序中，只包含两类文件：.cpp 文件和.h 文件。其中，.cpp 文件被称作 C++ 实现文件，其内容是 C++ 的源代码；.h 文件被称作 C++ 头文件，其内容主要是 C++ 的常量定义、类型定义和函数声明。C++ 实现文件主要保存函数的实现和类的实现。

1.2.1　头文件

通常，每一个.cpp 文件都有一个对应的.h 文件，也有一些例外，如单元测试代码只包含 main()的.cpp 文件，如例 1-1 所示。

1. 头文件举例

【例 1-1】 启动界面的头文件 start.h。

```
1. /*************************************************
2. Copyright (C), 2014-2015, 公司名称 Tech. Co., Ltd.
3. File name: start.h
4. //作者、版本及完成日期
5. Author: 初耀军    Version: 1.0v    Date:2015 年 12 月
6. Description:
7. 本模块实现注册和登录实现界面
8. Others:
9. 调用函数: menu_promt()
10. Function List:
11. menu_promt()
12. History:
13. Date:
14. Author:
15. Modification:
16. *************************************************/
17. #pragma once
18. #ifndef HEAD_START_H_
19. #define HEAD_START_H_
20. /************************************************
21.   Function:PromtMenu
22.   Description: 选择菜单提示
23.   Calls:  无
24.   Called By: 被函数 Register 和 Login 调用
25.   Table Accessed:
26.   Table Updated:
27.   Input:  无
```

第 1～16 行为注释。该注释是头文件的头注释，是必须的。但其内容，在不同的企业规范中不同。本注释以标识符“/*”开始，遇到行结束标识“*/”终止。中间部分是注释内容，是程序开发者为增加程序的易读性等目标而加入的说明。

第 17 行，#pragma once 与第 18、19 两行同时应用，所起到的作用相同。在头文件的开始处加入该指令，能保证头文件被编译一次。

第 20～31 行是函数部分注释。其内容，在不同的企业规范中不同。也有分隔作用，便于阅读。

```
28.   Output:
29.   Return: 无
30.   Others:
31. ************************************************/
32. void PromtMenu();
33. /************************************************/
34. #endif //HEAD_START_H
```

第 32 行是函数 void menu_promt();声明。

第 34 行是 ifndef 的结束。

说明：

第 18、19、34 行，构成如下格式：

```
#ifndef<标识>X      //先测试<标识>是否被宏定义过
#define<标识>X
程序段 1            //若 X 没有被宏定义过，定义 X，并编译程序段 1
#endif
程序段 2            //若 X 已经定义过了则编译程序段 2 的语句，“忽视”程序段 1
```

其作用是避免下面错误：若在 h 文件中定义了全局变量，一个 C++ 文件包含同一个 h 文件多次，若不加 # ifndef 宏定义或 # pragma once，会出现变量重复定义的错误；若加了 # ifndef 或 # pragma once，则不会出现重复定义的错。

2. 头文件的作用

正确使用头文件可令代码在可读性、文件大小和性能上大为改观，如例 1-1 所示。

(1) 通过头文件来调用库功能。在很多场合，源代码不便(或不准)向用户公布，只要向用户提供头文件和二进制的库即可。用户只需按照头文件中的接口声明来调用库函数，而不必关心接口是怎么实现的。连接器会从库中提取相应的代码，并和用户的程序链接生成可执行文件或者动态链接库文件。

(2) 头文件能加强类型安全检查。若某个接口被实现或被使用时的方式与头文件中的声明不一致，编译器就会指出错误，大大减轻程序员调试、改错的负担。

(3) 头文件可提高程序的可读性(清晰性)。

“头文件”不是编译单元，像 C++ 标准库一样，主要以“头文件”形式提供的库，只需让编译能找到这些头文件即可；而对于另外一些，直接以“实现文件”形式的扩展库，若要在某个项目中使用它，就必须提供“实现文件”，最好先复制一份，然后加入项目文件，参与编译。

3. 头文件的构成

头文件中的元素比较多，其顺序(结构)一般应安排如下，如例 1-1 所示。

(1) 头文件注释(包括文件说明、功能描述、版权声明等)(必须有)；

(2) 内部包含的卫哨开始(# ifndef XXX/ # define XXX)(必须有)；

(3) # include 其他头文件(若需要)；

(4) 外部变量和全局函数声明(若需要)；

(5) 常量和宏定义(若需要);

(6) 类型前置声明和定义(若需要);

(7) 全局函数原型和内联函数的定义(若需要);

(8) 内部包含的卫哨结束:#endif//XXX(必须有)。

上述排列顺序并非绝对,也无对错之分,可根据具体情况灵活安排。

若程序还需要内联函数,则内联函数的定义应当放在头文件中,因内联函数调用语句最终被扩展而不是采用真正的函数调用机制。

4. #ifndef…#endf

头文件通常包含了一些数据声明、函数声明、类型定义(如 struct/class)等内容,一个头文件通常要被多个实现文件包含。头文件之间可相互包含,这可能出现相同头文件往往会在同一个项目中被重复包含。对 C++ 来说,重复的包含一个头文件,不仅造成编译速度降低,还会带来编译错误:数据、函数声明允许重复,但一个类型不允许重复定义。

用#define 定义一个"宏符号",用#ifndef 来判断一个"符号"是否已经定义,如例 1-1 中对它们的使用,其格式如下:

```
#define HEAD_START_H_     //定义一个宏符号:名为 HEAD_START_H_
#ifdef HEAD_START_H_      //判断 HEAD_START_H_是否"已定义"
  /*
    这里的代码,仅当 start.h 有定义才会接受编译,否则被直接略过
  */
#endif
```

预编译器在处理"start.h"时,首先遇到下面这行预处理指令(通常是第一行):

```
#pragma once
```

只要在头文件的最开始加入它,就能够保证头文件只被编译一次。然后遇到:

```
#ifndef HEAD_START_H_
```

它判断宏符号"HEAD_START_H_"是否"未定义",因现在是第一次包含本文件,故确实还没有定义过它,于是下一行立即定义这个符号,然后才是本头文件的实质内容。

假设某一处的代码再次包含了这个头文件,但预编译器发现符号 HEAD_START_H_ 已经定义过了,于是它直接跳到#endif 之后。

通常为每个文件取一个唯一的宏符号,称为"保护符"。要使每一个头文件都有一个唯一的保护符,一般采用使该符号的名字和头文件名字有一个映射关系,习惯上,将所有字母都改成大写,再把扩展名之前的'.'改成'_',还可在前后分别再加一个下划线。通常,加上路径名以区别位于不同目录下的两个同名头文件。

5. 头文件的头注释

说明性文件的头部应进行注释,注释必须列出版权说明、版本号、生成日期、作者、内容、功能、与其他文件的关系等,头文件的注释中,还应有函数功能简要说明。

1.2.2 实现文件

1. 实现文件举例

【例 1-2】 启动界面。本启动界面为命令提示符状态下的启动界面，不是图形界面。主要显示图 1-2 中的注册、登录和退出组成的菜单。同时为使界面美观，用字符组成图形做修饰。运行结果，如图 1-1 所示。启动界面设计的 UML 活动图如图 1-2 所示。

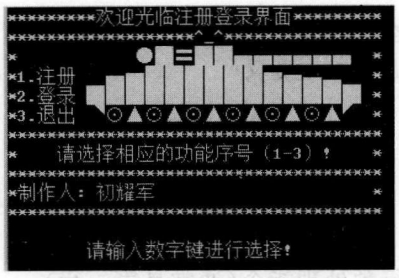

图 1-1 启动界面

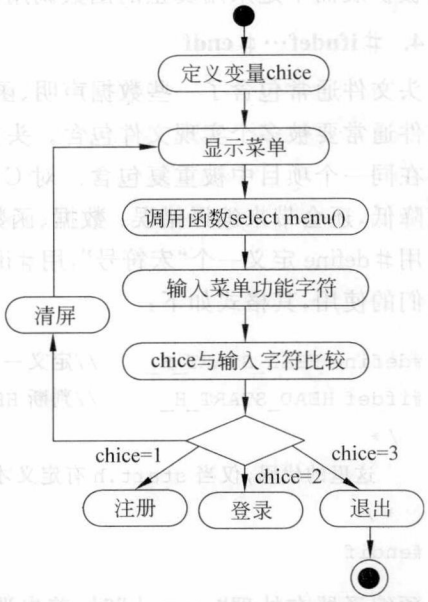

图 1-2 启动界面

设计思路：

(1) 在计算机的屏幕上显示字符，如没有特殊处理，显示字符的一般顺序是：从上到下、从左到右。

(2) 首先用固定不变的方式显示图 1-1 中的信息。为使示例具有 C++ 程序结构的一般性，又不复杂，将“请输入数字键进行选择!”通过调用另一个函数显示。

(3) 在输入功能序号(即输入数字字符 1～3)后，执行相应的操作，输入其他字符后回到启动界面。此处在程序中定义一个变量(这里的变量可暂时理解为数学中的变量) choice 表示功能序号。

① 当 choice 为 1 时，进入“注册”界面，此时这里没有实际内容，可以理解为预留的接口，在后面的设计中，再添加相应语句来完成。

② 当 choice 为 2 时，进入“登录”界面，思路与注册相同。

③ 当 choice 为 3 时，退出，即结束程序的运行，回到操作系统或调试界面，此处因使用 return 0，则返回调用该函数处，这里返回到操作系统。

④ 当 choice 取其他值时，重新显示启动界面。为显示的界面清晰，首先清理屏幕，再显示启动界面内容。清屏由语句“system("cls");”实现，其含义：调用 DOS 操作系统中

的清屏命令“cls”清屏，光标回到屏幕左上角。

(4) 多次重复显示界面，不可能多次重复书写或输入本例中的40～51行代码，采用类似数学递推的循环来设计。

程序代码，start.cpp文件：

```
1. /************************************************
2.   Copyright (C), 2014-2015,公司名称 Tech. Co., Ltd.
3.   FileName: start.cpp
4.   Author:初耀军 Version :1.0V    Date:2014年12月
5.   Description: 实现注册和登录项目的界面
6.   Version: 1.0V
7.   Function List:
8.   1. main()
9.   History:
10.      <author>  <time>  <version>  <desc>
11.       初耀军    2014/12   1.0      项目入口
12.   2. menu_promt()
13.   History:
14.      <author>  <time>  <version>  <desc>
15.       初耀军    2014/12   1.0      菜单选择
16.   3.Register()
17.   History:
18.      <author>  <time>  <version>  <desc>
19.       初耀军    2014/12   1.0      菜单选择
20.   4.Login()
21.   History:
22.      <author>  <time>  <version>  <desc>
23.       初耀军    2014/12   1.0      菜单选择
24. ************************************************/
25. #include<string>
26. #include<iostream>
27. using namespace std;
28. //#include "Register.h"
29. //#include "Login.h"
30. #include "start.h"
31.
32. /************************************************
33.   Function: main
34.   Description: 选择菜单
35.   Calls:  入口函数
36.   Called By: 调用函数 Register、Login 和 menu_promt
37.   Table Accessed:
38.   Table Updated:
39.   Input:
```

第1～24行头文件的注释，必须有。其内容，在不同的企业规范中不同。

第25行包含string头文件，后面的代码会需要string中的函数等。

第26行预处理包含指令，它通知编译器将类库的标准输入输出流头文件iostream包含到程序中。iostream文件提供了输入和输出流cin和cout及输入输出运算符>>和<<定义。

第27行using namespace std。所有的标准库函数都在标准命名空间std中进行定义的。以避免发生重命名。

第28～30行预处理包含指令，它通知编译器将相应的头文件包含到程序中，因程序后面代码需要。此处主要是start.h，但第28～29行被注释，为后面的应用做准备。

第32～44，89～101行函数的注释，必须有。不同的企业的规范不同。也有分隔作用，便于阅读。第45～88、102～105行是不同函数的定义。它们的首部依次为int main()、void menu_promt()。

以int main()为例。其中int是该函数的返回类型，main是函数名，一对括号“()”里面，放置该函数的形