21世纪高等教育计算机规划教材

COMPUTER

C++ 程序设计实践案例教程

Practice of C++ Programing Tutorial

朱晓凤 卢青华 陈鑫 王红勤 编著

张屹 王刚 主审

- 针对 C++ 语言面向对象的内容编写
- 基于实际教学，将理论与实践结合
- 案例丰富，学以致用

人民邮电出版社
POSTS & TELECOM PRESS

21世纪高等教育计算机规划教材

COMPUTER

C++ 程序设计实践案例教程

Practice of C++ Programing Tutorial

朱晓凤 卢青华 陈鑫 王红勤 编著

张屹 王刚 主审

人民邮电出版社

北京

图书在版编目（CIP）数据

C++程序设计实践案例教程 / 朱晓凤等编著. -- 北京 : 人民邮电出版社, 2015.1
21世纪高等教育计算机规划教材
ISBN 978-7-115-38132-3

Ⅰ. ①C… Ⅱ. ①朱… 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(2015)第007867号

内 容 提 要

本书是针对C++程序设计的相关课程编写的，从对象和类的角度来安排内容，共分为13个项目，其中10个项目是分别对应每个知识点的实践案例；另外3个项目是综合项目案例。每一个项目都包括基础知识、案例实训、习题及解析等部分。在每个项目案例里，都给出了例题和参考解答方法，然后提出思考题，让读者在模仿的基础上思考，从而提高学生的程序设计能力。

本书可作为高等院校计算机、软件工程专业本科生的教材，同时可供C++语言的初学者使用。

◆ 编　　著　朱晓凤　卢青华　陈　鑫　王红勤
主　　审　张　屹　王　刚
责任编辑　许金霞
责任印制　沈　蓉　彭志环

◆ 人民邮电出版社出版发行　　北京市丰台区成寿寺路11号
邮编　100164　　电子邮件　315@ptpress.com.cn
网址　http://www.ptpress.com.cn
三河市潮河印业有限公司印刷

◆ 开本：787×1092　1/16
印张：11.5　　2015年1月第1版
字数：301千字　　2015年1月河北第1次印刷

定价：29.00元

读者服务热线：(010)81055256　印装质量热线：(010)81055316
反盗版热线：(010)81055315

前言

C++程序设计是软件开发等专业方向必修的程序设计基础课程之一。本书通过介绍 C++语言中面向对象编程技术，使学生了解高级程序设计语言的结构，掌握基本的程序设计过程和技巧，掌握基本的分析问题和利用计算机求解问题的能力，具备初步的高级语言程序设计能力，为以后的面向对象语言学习打好基础。

在学习 C++语言之前，读者应该已经学习过 C 语言编程，对编程有了一定的了解。C++是一种非常复杂的语言，所涵盖的内容非常广泛。有效使用这一语言最重要的关键是理解 C++各特性之间的互动关系。本课程的学习可使学生掌握 C++面向对象程序设计的基本概念和基本方法，掌握如何利用已经学会的工具来解决问题，也就是编程的方法论，培养学生以面向对象方法来分析、解决实际问题的能力，为后续课程的学习打下良好的基础。

C++语言是一种非常重要的编程语言，也是一门实践性很强的课程，因此实践实验是掌握这门语言的重要途径。对于学习者而言，一本具有丰富案例和详细实验指导的书是必不可少的。希望读者明白，语言只是工具，重要的是如何用这个工具解决你需要处理的问题，而不是把主要精力放在如何全面仔细地对工具进行研究。

本书是按照 C++各知识点的学习过程，并基于实践案例来进行编写的，所以每一部分都是以项目的方式编排的。

本书总共分为 13 个项目，其中 10 个项目是分别对应每个知识点的实践案例，另外 3 个项目是综合项目案例。每一个项目都包括基础知识、案例实训、习题及解析等部分。每一章节的内容包括知识点补充、习题、实验和思考题等部分，让学习者从各个方面加深对 C++语言的学习。

本书的特点在于，在每个项目案例里面，均给出了例题和参考解答方法，然后提出思考题，让读者在模仿的基础上思考，进而写出具有自己风格的代码。

本书在编写过程中得到了很多老师的帮助。其中在框架结构和内容安排上，张屹老师提出了建设性的意见。另外，王刚老师帮助联系出版社，并督促本书的编写进度。在此向他们表示衷心的感谢。

本书的编者有朱晓凤、卢青华、陈鑫和王红勤。在编写过程中，参考了大量文献资料，书中若有疏漏与不当之处，敬请广大读者批评指正。

编　者

2014 年 11 月

目　录

项目 1 绪论

1.1 基础知识

1.1.1 基本概念

1. 面向过程与面向对象

C++语言作为当前主流的软件开发语言，包括过程性语言和类部分，也就是面向过程部分和面向对象部分，过程性语言部分与 C 并无本质的差别，C++语言完全兼容 C 语言。

“面向过程”是一种以过程为中心的编程思想，也可称之为“面向记录”编程思想，它们不支持丰富的“面向对象”特性（比如继承、多态），并且它们不允许混合持久化状态和域逻辑。面向过程程序设计是指分析出解决问题所需要的步骤，然后用函数把这些步骤一步一步实现，使用的时候一个一个依次调用就可以了。

“面向对象”（Object Oriented，OO）是一种以事物为中心的编程思想。面向对象的程序设计（Object-Oriented Programming，OOP）是指在程序设计中采用一种对现实世界理解和抽象的方法，“面向对象”是专指在程序设计中采用封装、继承、多态等设计方法。

2. Eclipse 平台

Eclipse 平台是一个开放源代码的、基于 Java 的可扩展开发平台。就其本身而言，它只是一个框架和一组服务，用于通过插件组件构建开发环境。幸运的是，Eclipse 平台附带了一个标准的插件集，包括 Java 开发工具（Java Development Kit，JDK）。在客户机平台上，Eclipse 使用插件来提供所有的附加功能，例如已经能够支持 C/C++（CDT）插件。

虽然大多数用户很乐于将 Eclipse 平台当作 Java 集成开发环境（IDE）来使用，但 Eclipse 平台的目标却不仅限于此。Eclipse 平台还包括插件开发环境（Plug-in Development Environment，PDE），这个组件主要针对希望扩展 Eclipse 平台的软件开发人员，因为它允许他们构建与 Eclipse 平台环境无缝集成的工具。由于 Eclipse 平台中的每样东西都是插件，对于给 Eclipse 平台提供插件以及给用户提供一致和统一的集成开发环境而言，所有工具开发人员都具有同等的发挥场所。

这种平等和一致性并不仅限于 Java 开发工具。尽管 Eclipse 是使用 Java 语言开发的，但它的用途并不限于 Java 语言，例如，支持诸如 C/C++、COBOL、PHP 等编程语言的插件已经可以使用，或预计将会推出。Eclipse 框架还可用来作为与软件开发无关的其他应用程序类型的基础，比

如内容管理系统。

Eclipse 的设计思想是：一切皆插件。Eclipse 核心很小，其他所有功能都以插件的形式附加于 Eclipse 核心之上。Eclipse 基本内核包括：图形 API（SWT/Jface、Java 开发环境插件（JDT）、插件开发环境（PDE）等。

Eclipse 的插件机制是轻型软件组件化架构。在客户机平台上，Eclipse 使用插件来提供所有的附加功能，例如支持 Java 以外的其他语言。已有的分离的插件已经能够支持 C/C++（CDT）、Perl、Ruby，Python、Telnet 和数据库开发。插件架构能够支持将任意的扩展加入现有环境中，例如配置管理，而决不仅仅限于支持各种编程语言。

CDT（C/C++ Development Tooling）是 Eclipse IDE 的一个重量级插件项目，提供了一个功能齐全的 C 和 C++集成开发环境（IDE）。功能包括：支持项目的创建和管理的各种工具链、标准 make build、代码导航、各种代码知识工具，如类层次结构、调用图，包括代码浏览器、宏定义的浏览器、代码编辑器支持语法高亮、折叠和超链接导航、源代码重构和代码生成、可视化调试工具，包括内存、寄存器、反汇编浏览器等，非常适合构建各种规模（特别是大型）的 C/C++项目。

使用本实训环境调试程序的一般步骤为：创建项目，添加文件，编写代码，调试，运行。

1.1.2 编程规范

1. 排版

程序采用缩进风格编写，缩进的空格数为 4 个，不使用 Tab 键；程序块的分界符应一个独占一行并且位于同一列，同时与引用它们的语句左对齐；

if，for，do，while，case，switch，default 等语句各自占一行，且 if，for，do，while，case 等语句的执行语句部分无论多少都要加“{}”；

相对独立的程序块之间、变量声明语句块之后必须加空格；较长的语句（>80 字符）要分成多行书写，长表达式要在低优先级操作符处划分新行，操作符放在新行之首，划分出的新行要进行适当的缩进，使排版整齐，语句可读；

不允许把多个语句写在一行中，即一行只写一条语句；不允许把多个变量写在一行中，即一行只定义一个变量；

在两个以上的关键字、变量、常量进行对等操作时，它们之间的操作符之前、之后或前后加空格；进行非对等操作时，如果是关系密切的立即操作符（如->）后不应加空格；

判断语句中常量/宏放在“==”的左边，变量放在“==”的右边；

访问权限的编译开关和控制关键字 public/private/protected 与上一级代码保持对齐，不缩进。

2. 注释

说明性文件头部应进行注释，注释必须列出：版权说明、生成日期、作者、内容说明、修改日志等；

源文件头部应进行注释，列出：版权说明、生成日期、作者、模块目的/功能、主要函数、修改日志等；

修改代码同时修改相应的注释，以保证注释与代码的一致性，不再有用的注释要删除；

变量名、常量名、数据结构名（包括数组、结构、类、枚举等）如果不是充分自注释的，则必须加上注释；

较复杂的分支语句（条件分支、循环语句等）必须加上注释；

对于 switch 控制块中不含 break 的 case 语句，必须加上注释；

对于使用代码屏蔽警告的，必须加上注释，至少列出屏蔽原因、批准人员、批准时间；

注释应与其描述的代码位置相近，对代码的注释通常放在其上方相邻位置，并与注释的代码采用相同的缩进，不可放在下面和语句中间，如放于上方，则需与其上面的代码用空行隔开；

对于变量名、常量名、数据结构子元素等单行语句，其注释语句可放在右边；

注释的内容要清楚、明了，含义准确，防止注释二义性；

注释中禁止使用缩写，除非已是业界通用或标准化的缩写；

源程序注释量必须达到 20%以上。

3. 命名规则

标识符的命名必须清晰、明了，有明确含义；

命名中禁止使用缩写，除非已是业界通用或标准化的缩写；

禁止使用单个字符作为变量名（包括用于循环的临时变量），要使用有意义的单词；

除非必要，不要用数字或较奇怪的字符来定义标识符；

用正确的反义词组命名具有互斥意义的变量或相反动作的函数等；

除用于编译开关和防止头文件重复展开的宏外，禁止名字以下划线起始或结尾。

1.2　实训——开发环境的搭建与简单程序实现

1.2.1　实训目的

1. 了解 Eclipse 平台的框架。
2. 熟悉基于 Eclipse 平台的 C++程序开发环境的搭建过程。
3. 熟悉开发环境的各个菜单命令的位置。
4. 掌握开发程序的步骤和运行调试的过程。
5. 熟悉新建项目、编辑、编译、运行、调试代码。

1.2.2　实训内容与步骤

1. 搭建实训环境

说明：当前编译 C++语言的平台有很多，常见的有 Microsoft Visual C++、Visual Studio 和 Borland C++等。近年来，随着 Java 语言的发展，Eclipse 平台也迅速发展起来，它是一个开放源代码的、基于 Java 的可扩展开发平台。

在本书中会介绍基于 Eclipse 平台的 C++开发环境的搭建方法，因为其他平台的搭建都十分容易。同时，本书中所给出的代码在各个编译环境均可正常运行。

本书中的代码即是在此环境中进行开发的，需要安装相关的软件，这里我们选取安装 jdk-1.6-windows-i586.exe 和 eclipse-cpp-ganymede-SR2-win32（32 位 windows 操作系统）。

（1）安装 JDK。我们可以在 http://java.sun.com/j2se/1.5.0/download.jsp 上下载并安装。安装完成之后，在 DOS 界面中测试安装是否成功。在命令行中输入 java 命令和 javac 命令，效果分别如图 1-1 和图 1-2 所示。

（2）有两种选择，可以先安装 Eclipse IDE，再安装 CDT；也可以直接安装集成 CDT 的 Eclipse。软件的获取地址为 http://eclipse.org。

图 1-1　java 测试效果图

我们这里选择第二种方式，即下载并安装 eclipse-cpp-ganymede-SR2-win32，它的版本是 3.4，也叫做 ganymede（木卫三）。下载软件时注意软件和操作系统的兼容性，这里选择的是 32 位 Windows 操作系统中适用的版本，具体的下载地址为 http://eclipse.org/downloads/ packages/release/Ganymede/SR2。

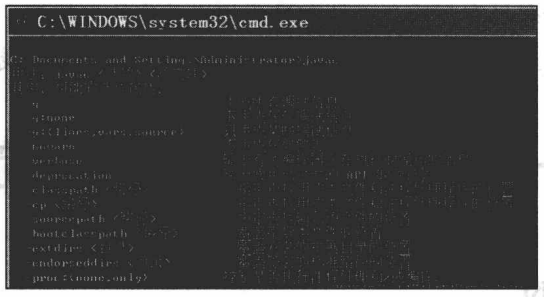

图 1-2　javac 测试效果图

这个安装十分简单，只需要把文件解压缩之后拷贝到特定目录下即可。安装完成之后，运行程序，运行后一个灰蓝色的 Welcome 页面出现，进入 Tutorials（教程）。学东西先读 Tutorial 是个好习惯，如图 1-3 所示。

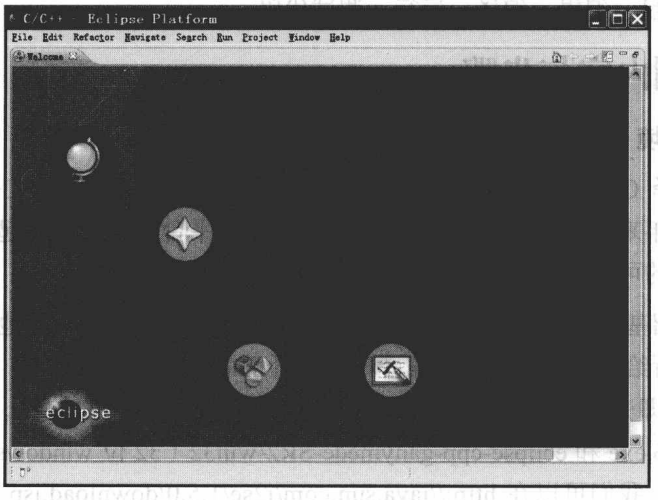

图 1-3　Eclipse 界面图

（3）安装 MinGW。到目前为止，已经为 Eclipse 装上了 CDT，但还需要一个“编译程序”才能编译程序，因此要下载可在 Windows 上使用的 GNU C、C++编译程序，这里要下载的是 MinGW。MinGW 是 Minimal GNU for Windows 的意思，我们可以在 http://www.mingw.org/download. shtml 网站上面获取这个软件。

安装完成之后，来测试是否安装成功，在 DOS 界面中输入命令：gcc –v，如果成功会看到如图 1-4 所示的 gcc 命令测试图版本信息。如果没有得到这个结果，那么就需要修改环境变量，例如我们把 MinGW 存放在 C 盘根目录下，那么在 path 中增加路径：C:\MinGW\bin。

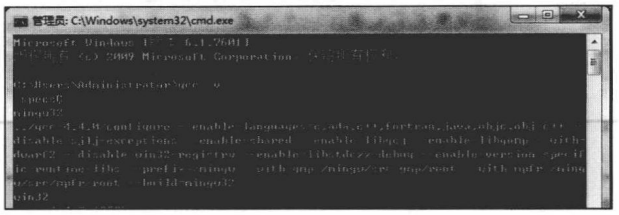

图 1-4　gcc 命令测试图

（4）安装 gdb debugger。如果你需要断点调试程序，还需要安装 gdb debugger，这里可以选择 gdb-6.6，地址为 http://downloads.sourceforge.net/mingw/gdb-6.6.tar.bz2。解压 gdb-6.6.tar.bz2 到你安装 MinGW 的地方，gdb-6.6 文件目录下也有一系列 bin，inclue 文件夹，直接拷贝到 MinGW 下面覆盖进去即可。

（5）设置 Eclipse 参数。为了使 CDT 能够取用 MinGW 来进行编译工作，我们要回到 Eclipse 当中进行设定：Window→Preferences→C/C++→New CDT Project Wizard→Makefile Project，找到 Binary Parser，取消 Elf Parser，改选 PE Windows Parser。

（6）创建项目并运行。运行 Eclipse，选择 File→New→C++ Project，弹出 C++ Project 对话框，在 Project name 中输入项目名称 lab1_1，如果想要修改项目的保存目录，可以把 Use default location 复选框的打勾号取消，然后在下面输入保存路径或者单击 Browse 按钮，选择保存路径；否则，保存在默认的目录里面。在 Project type 里面 Executable 选项里选择 Hello World C++ Project 选项，可以省去自己配置开发环境的过程。在右边的 Tollchains 中选择 MinGW GCC，然后单击 Next 或者 Finish 进入下一个对话框或者直接进入开发环境，如图 1-5 所示。

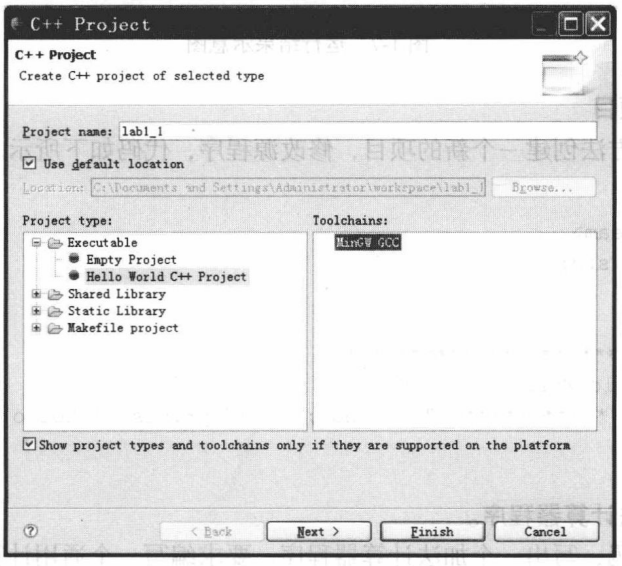

图 1-5　创建项目图

在出现的界面左边 Project Explorer 下单击刚刚创建的项目名称，出现一个树形结构，可以看到 src 子目录，在其下面已经生成了一个源文件，我们就可以在该源文件的基础上进行修改，如图 1-6 所示。

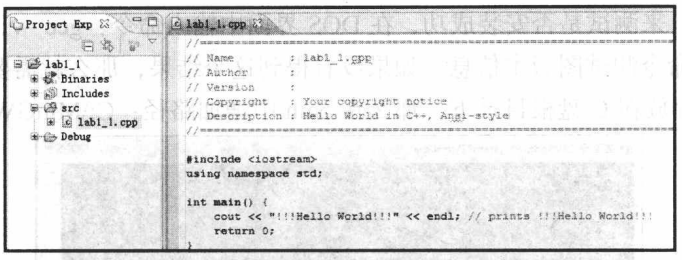

图 1-6　调试界面示意图

当对项目进行编辑之后，选择 Project→Build Project 对该项目进行编译链接，如果没有错误，则选择菜单 Run→Run As→Local C/C++ Application，程序就会运行起来。我们可以在屏幕的下方 Console 中看到程序运行的输出结果，如图 1-7 所示。

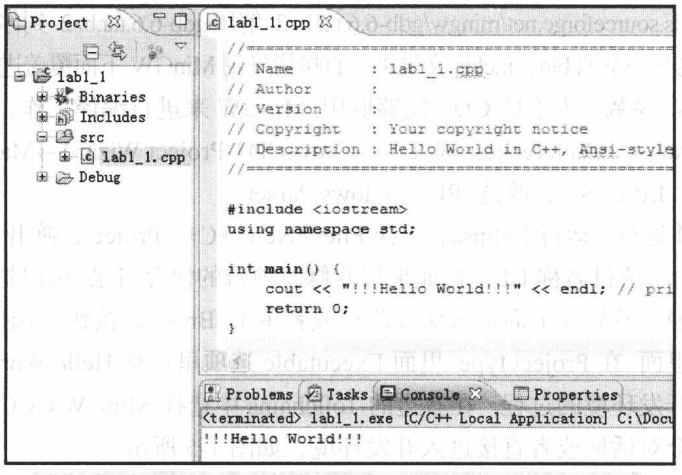

图 1-7　运行结果示意图

2. 学习创建新项目

按照前面所说的方法创建一个新的项目，修改源程序，代码如下所示。运行程序，查看并解释运行结果。

```
#include <iostream>
using namespace std;

int main() {
    cout <<"********************\n\n"
     << "*   Hello World   *\n\n"
     <<"********************\n" << endl;     // prints !!!Hello World!!!
    return 0;
}
```

3. 编写一个加法计算器程序。

仿照上一题的代码，写出一个加法计算器程序。要求编写一个通用计算器程序，当用户输入两个数以后，可以计算并输出这两个数的和、差、积、商。

4. 编写一个计算直角三角形斜边边长的程序。

编写一个计算直角三角形斜边边长的程序，要求当输入直角三角形的两个直角边边长后，能够计算出第三边（即斜边）的边长。

解析：由直角三角形的直角边计算斜边的代码格式为：sqrt(a*a+b*b)，其中 a,b 为两条直角边。参考代码如下所示。

```
#include <iostream>
#include<math.h>
using namespace std;

int main() {
    int a,b;
    cout <<"请输入直角三角形的两个直角边长："<<endl;
    cin>>a>>b;
    cout<<"直角三角形的斜边长："<< （1） <<endl;
}
```

思考：如果在程序中需要检测构成三角形的条件，以及检测构成直角三角形的条件，该如何修改代码？是否需要引入异常处理？

1.2.3　实训总结

通过自己动手，掌握实训环境的搭建，了解开发环境所需的软件包以及相关的环境配置，掌握创建、编辑、修改、编译、运行程序的方法和步骤，能够编写并运行简单的程序。

1.3　习题及解析

一、选择题

1. 对 C++语言和 C 语言的兼容性描述正确的是：（　　）。

A. C++兼容 C　　B. C++部分兼容 C　　C. C++不兼容 C　　D. C 兼容 C++

2. 下列关于 C++语言的发展说法错误的是（　　）。

A. C++语言起源于 C 语言　　B. C++语言最初被称为“带类的 C”

C. 在 1980 年 C++被命名　　D. 在 1983 年 C++被命名

3. C++语言是以（　　）语言为基础逐渐发展演变而成的一种程序设计语言。

A. Pascal　　B. C　　C. Basic　　D. Simula67

4. 下列关于 C++与 C 语言关系的描述中错误的是（　　）。

A. C++是 C 语言的超集

B. C++是 C 语言进行了扩充

C. C++和 C 语言都是面向对象的程序设计语言

D. C++包含 C 语言的全部语法特征

5. 下列 C++标点符号中表示行注释开始的是（　　）。

A. #　　B. ;　　C. //　　D. }

6. 每个 C++程序都必须有且仅有一个（　　）。

A. 预处理命令　　B. 主函数　　C. 函数　　D. 语句

7. C++对 C 语言做了很多改进，下列描述中哪一项使得 C 语言发生了质变，即从面向过程变成面向对象：（　　）。

A. 增加了一些新的运算符　　B. 允许函数重载，并允许设置默认参数

C. 规定函数说明必须用原型　　D. 引进类和对象的概念

8. 下列不正确的选项是（　　）。

A. C++语言是一种既支持面向过程程序设计，又支持面向对象程序设计的混合型语言

B. 关键字是在程序中起分割内容和界定范围作用的一类单词

C. iostream 是一个标准的头文件，定义了一些输入输出流对象

D. 类与类之间不可以进行通信和联络

9. 根据编码规范，以下说法不正确的是（　　）。

A. 每行中只能写一条赋值语句

B. 若 a 为实型变量，C++程序中允许赋值 a=10，可以把整型数赋给实型变量

C. 无论是整数还是实数，都能被准确无误地表示

D. %是只能用于整数运算的操作符

10. 根据编程规范，一般情况下，源程序有效注释量必须在（　　）。

A. 20%以上　　B. 20%以下　　C. 10%以上　　D. 10%以下

二、填空题

11. _____就是某一变量的别名，对其操作与对变量直接操作完全一样。

12. 按函数在语句中的地位分类，可以有以下 3 种函数调用方式：_____，_____，_____。

13. 函数与引用联合使用主要有两种方式：一是_____；二是_____。

14. 头文件由三部分内容组成：头文件开头处的文件头注释，_____，_____。

15. 用于输出表达式值的标准输出流对象是_____。

16. 用于从键盘上为变量输入值的标准输入流对象是_____。

17. 一个函数定义由_____和_____两部分组成。

18. C++头文件和源程序文件的扩展名分别为_____和_____。

19. C++既可以用来进行面向_____程序设计，又可以进行面向_____程序设计。

20. 常量分成两种，一种是_____常量，另一种是_____常量。

参考答案

1～5. A C B C C　　6～10. B D D C A

11. 引用　　12. 函数语句　函数表达式　函数参数

13. 函数的参数是引用　函数的返回值是引用　　14. 预处理块　函数和类结构声明

15. cout　16. cin　　17. 函数头　函数体　　18. .h　.cpp

19. 对象　过程　　20. 直接　符号

1.4 思考题

1. 开发程序中的相关操作，除编辑、修改、编译、调试、运行之外，是否还有其他方法？

2. 开发环境中的一些常用的快捷键有哪些？

项目 2 类和对象

2.1 基础知识

2.1.1 类和对象

1. 面向对象编程设计

面向对象编程（Object Oriented Programming，OOP）是一种计算机编程架构。它实现了软件工程的 3 个主要目标：重用性、灵活性和扩展性。在程序中，为了达到整体运算的目的，要求每个对象都能接收信息，并且处理数据，然后向其他对象发送信息。

在程序设计中，不适用面向过程的程序设计思想，而采用面向对象的编程思想，好处有：

（1）当需求发生变化时，程序的主函数 main 的代码可以不需要做任何修改，更容易维护和扩充。

（2）书写的代码比其他形式的表示更易于理解。

（3）程序比其他形式实体有更好的可修改性。

（4）可以增强程序的可读性，使得程序更为简明、清晰。

（5）对于强类型的 C++语言，更容易检查非法使用，使得编译程序能够根据类型定义与使用之间的关系。

面向对象程序设计中的概念主要包括类、对象、数据封装、继承、多态性、消息传递等，通过这些概念面向对象的思想得到了具体的体现。

（1）类（class）：一个共享相同结构和行为的对象的集合。

（2）对象（object）：类中的一个具体个体，对应到现实世界中一个个体，它有状态、行为和标识 3 种属性。

（3）封装（encapsulation）：将数据和行为（算法）捆绑在一起，定义出一个新的数据类型的过程。

（4）继承：代码重用的一种方法，描述了类之间的一种关系，在这种关系中，一个类共享了一个或多个其他类定义的结构和行为。继承描述了类之间的“是一个”关系。子类可以对基类的行为进行扩展、覆盖、重定义。

（5）多态：在继承的情况下，子类和派生类对同一个消息表现出不同的行为，这被称为动态多态性。

（6）消息传递：一个对象调用了另一个对象的方法（或者称为成员函数），简单来说就是函数调用。

2. 类和对象

（1）类的声明

事物的静态特征可以用某种数据来描述，一般用数据成员表示事物的静态特征，描述事物（对象）所表现的行为或具有的功能，一般用成员函数表示。其形式声明为：

```
class  类名称
{
  public:
          公有成员（外部接口）
  private:
          私有成员
  protected:
          保护型成员
};
```

类包含两种不同的成员：① 数据成员；② 成员函数。

（2）对象的定义

类名　对象名；

类名　对象名 1，对象名 2，对象名 3…；

类名　* 对象指针名；

类名　对象数组名[成员个数]；

（3）成员函数

类的成员函数描述的是类的行为，是对封装的数据进行操作的方法。成员函数根据所在位置来分类，可以分为在类内部定义（内联函数）和在类外部定义。

成员函数的调用一般有两种方法：

① 对象名.函数名（实参 1，实参 2，…）

② 对象指针→函数名（实参 1，实参 2，…）

（4）成员的权限

私有成员（private）：可以被类自身的成员和友元访问，其他不可以。

保护成员（protected）：可以被类自身的成员和友元访问，其他不可以。

公有成员（public）：可以被类自身的成员和友元访问，也可以被对象任意访问。

2.1.2　函数

在程序设计中，函数是必不可少的，它体现了模块化的程序设计思想。函数就是功能，每一个函数用来实现一个特定的功能，函数的名称一般能反映这个函数的功能。设计较大程序的时候，往往会分成不同的模块，每个模块用一个函数来实现。在编程规范中，每个程序块的长度一般不超过一个屏幕的长度或者不超过 15 行。

函数定义的一般形式为：

```
函数类型 函数名(类型名 形式参数 1,… )
{
  说明语句
  执行语句
}
```

函数调用是多种多样的，常见的几种调用方式有：

（1）函数调用作为语句被调用；

（2）表达式中的函数调用；

（3）函数调用作为实参被调用。

1. 参数传递

函数的参数传递指的是在程序运行过程中，实际参数就会将参数值传递给相应的形式参数，然后在函数中实现对数据处理和返回的过程，方法有按值传递参数，按地址传递参数和按引用传递参数。在C++中调用函数时，有3种参数传递方式：

（1）传值调用；

（2）传址调用（传指针）；

（3）引用传递。

前两者为值传递，最后一种为引用传递。

按引用传递，引用实参的引用参数传递给函数，而不是进行参数拷贝。引用类型的形参与相应的实参占用相同的内存空间，改变引用类型形参的值，相应实参的值也会随着变化。

2. 函数的重载

重载函数（overloaded function）是C++支持的一种特殊函数，C++编译器对函数重载的判断更是C++语言中最复杂的内容之一。在相同的声明域中的函数名相同而参数表不同的，即是通过函数的参数表而唯一标识并且来区分函数的一种特殊的函数。

函数的重载其实就是“一物多用”的思想，形式上调用同一个函数名，实现的功能却不同。其实不仅是函数可以重载，运算符也是可以重载的。例如：运算符“<<”和“>>”既可以作为移位运算符，又可以作为输出流中的插入运算符和输入流中的提取运算符。

函数重载中的参数的个数和类型可以都不同。但不能只有函数的类型不同而参数的个数和类型相同。

两个重载函数必须在下列一个或两个方面有所区别：

（1）函数有不同参数；

（2）函数有不同参数类型。

C++的这种编程机制给编程者极大的方便，不需要为功能相似、参数不同的函数选用不同的函数名，也增强了程序的可读性。

函数重载的原则：自动数据转换对函数重载的影响，两个不同宽度的数据类型进行运算时，编译器会尽可能地在不丢失数据的情况下将它们的类型统一；在进行float和double运算时，如果不显式地指定为float型，会自动转换成double型进行计算；若一个整数类型int和一个浮点类型float运算时，如果不显式地指定为int型，C++会先将整数转换成浮点数。

3. 有默认参数的函数

C++允许在定义函数时给其中的某个或某些形式参数指定默认值，这样，当发生函数调用时，如果省略了对应位置上的实参的值时，则在执行被调函数时，以该形参的默认值进行运算。有时多次调用同一函数时用同样的实参，给形参一个默认值，这样形参就不必一定要从实参取值了。实参与形参的结合是按从左至右顺序进行的。因此指定默认值的参数必须放在形参表列中的最右端，否则出错。

在使用带有默认参数的函数时有两点要注意：

（1）如果函数的定义在函数调用之前，则应在函数定义中给出默认值。如果函数的定义在函数调用之后，则在函数调用之前需要有函数声明，此时必须在函数声明中给出默认值，在函数定