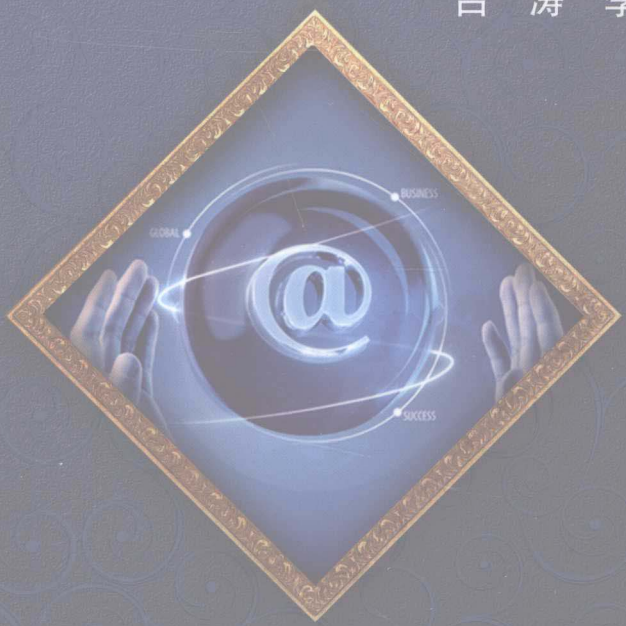
普通高等教育“十二五”规划教材

C++面向对象程序设计习题与实验指导（第二版）

张 俊 主 编
吕 涛 李晓林 参 编



教材资源网址：
http://www.51eds.com

中国铁道出版社
CHINA RAILWAY PUBLISHING HOUSE

普通高等教育“十二五”规划教材

C++面向对象程序设计习题与实验指导

（第二版）

张　俊　主编

吕　涛　李晓林　参编

中国铁道出版社
CHINA RAILWAY PUBLISHING HOUSE

内 容 简 介

《C++面向对象程序设计习题与实验指导（第二版）》与主教材《C++面向对象程序设计（第二版）》配套使用。

本书在第一版的基础上进行了全面修订，延续了第一版的篇章结构和内容体系，充实了各个部分的内容。全书包含三部分：第一部分总结提炼了主教材各章的重点难点和主要知识点，同时配以适量的测试题；第二部分为实验指导，针对每章的能力要求设计了适宜的例题和上机练习题；第三部分为 STL 算法参考与容器参考，旨在为学习 STL 提供方便快捷的参考。

本书适合作为计算机科学与技术及相关专业面向对象程序设计和 C++语言课程的教材，也可供读者自学和参考。

图书在版编目（CIP）数据

C++面向对象程序设计习题与实验指导/张俊主编. — 2 版. — 北京：中国铁道出版社，2012.8
普通高等教育“十二五”规划教材
ISBN 978-7-113-14630-6

Ⅰ. ①C… Ⅱ. ①张… Ⅲ. ①C 语言－程序设计－高等学校－教学参考资料 Ⅳ. ①TP312

中国版本图书馆 CIP 数据核字(2012)第 173020 号

书　　名：C++面向对象程序设计习题与实验指导（第二版）
作　　者：张　俊　主编

策　　划：杨　勇　　**读者热线：400-668-0820**
责任编辑：吴宏伟　彭立辉
封面设计：付　巍
封面制作：刘　颖
责任印制：李　佳

出版发行：中国铁道出版社（100054，北京市西城区右安门西街 8 号）
网　　址：http://www.51eds.com
印　　刷：三河市兴达印务有限公司
版　　次：2008 年 8 月第 1 版　2012 年 8 月第 2 版　2012 年 8 月第 2 次印刷
开　　本：787mm×1092mm　1/16　印张：18.25　字数：440 千
印　　数：5 001～8 000 册
书　　号：ISBN 978-7-113-14630-6
定　　价：33.00 元

版权所有　侵权必究

凡购买铁道版图书，如有印制质量问题，请与本社教材图书营销部联系调换。电话：（010）63550836

打击盗版举报电话：（010）63549504

第二版前言

FOREWORD ➤➤➤

本书第一版于2008年8月出版，经过几年的使用和教学实践，得到了许多反馈意见，在中国铁道出版社的支持下，于2012年2月进行了改版工作。本书延续了第一版的篇章结构和内容体系，同时对相关内容进行了较大幅度的修订，主要包括：

（1）重新编写了第一部分每章的知识点归纳部分，更新充实了各章的练习题。“知识点归纳”部分的目的是要让使用者能够更快地回忆本章的重要概念、方法和技巧。在修订过程中，重新归纳之后的知识点更加紧凑，重点和难点更加突出。所更新的习题参考了各类考试的风格和特点，选取了难度不同、题量适度的习题供学生巩固本章概念和方法。

（2）修订了第二部分实验指导中每章的示例及其源代码。程序调试是程序设计中的一个难点，基于C++类的程序更是具有较大的调试难度，修订之后的版本增加了程序调试的一些技巧和上机练习。同时，还对每章的例题和上机练习题进行了修订，对一些比较综合的上机练习题提供了编程提示。

（3）修订了第三部分有关 STL 的参考及其示例程序。学习程序设计语言要重视对其标准库的学习，学习C++语言更是如此。标准模板库（STL）是C++语言的重要组成部分，越来越多的用人单位对 STL 有着较高的期望，因此对这部分内容进行了修订充实，目的是让学生能够更熟练地应用STL。

编者衷心期望本书能够在培养学生实践能力和综合应用能力方面起到不可或缺的作用，使得学生更加热爱程序设计，更加注重通过应用软件技术解决生活中和学习上的一些问题。这样就达到了编写此辅教材的初衷。

在本书改版过程中，得到了江世宏老师、王庆春老师的关心和指导，同时感谢张彦铎教授、王海晖教授、王忠教授等的热情支持。

特别需要感谢中国铁道出版社的编辑，在本书两版的编辑过程中，他们一直保持着严谨而专业的精神、耐心及温和态度，令人甚是钦佩。最后，诚挚地感谢在本书出版过程中作出各种工作的人！

本书凝结了编者多年来在C++语言和面向对象程序设计教学实践中的经验，虽经改版，其间修订数次，但由于编者水平有限，仍难免存在疏漏和不足之处，诚恳期待并接受读者的批评和指正。

编　者

2012年2月

第一版前言

<<< FOREWORD

《C++面向对象程序设计》教材以C++语言作为载体，介绍了C++程序设计的基本概念、主要思想和常用算法，并以面向对象程序设计的思想和方法为重点，讲解了面向对象程序的主要内容和重要方法。该门课程对于培养计算机专业学生的计算机应用能力起着重要的基础作用。程序设计是一门尤其需要上机实践和实训的课程，为了更好地培养学生的程序设计能力和综合应用能力，并配合《C++面向对象程序设计》课程的教学需要，我们编写了《C++面向对象程序设计习题与实验指导》一书。

本书内容包括三个部分。第一部分为基础知识与习题，按照《C++面向对象程序设计》教材的体例结构，对应教材的每一章节，概括了各章的能力要求，分析了重点和难点，并细致的归纳了各章的知识点，随后配以适量的测试练习题，以便巩固和强化基本概念和知识重点。第二部分为实验指导，首先以实验目的与要求的形式强调了每章应重点掌握的关键知识点，并对实验过程与实验内容进行了详细讲解和说明，再辅以典型程序与示例详细讲解了主要知识的应用，最后布置了适量的实验题目。第三部分为STL算法和容器参考，重点介绍STL算法与容器。

本书中的示例程序和各种代码都在VC++ 2005环境下编译测试，各部分内容相互配合。恰当使用本书，对于面向对象程序设计和C++语言课程的学习具有重要的促进作用，对于分析能力、设计能力的提高不无裨益。

限于时间和编者的水平，书中疏漏和不妥之处在所难免，敬请读者批评指正。

编　者

2008年6月

此为试读，需要完整PDF请访问：www.ertongbook.com

目 录

CONTENTS >>>

第一部分 习 题

第二部分　实 验 指 导

第三部分　STL 算法与容器参考

第一部分　习　　题

第1章　C++语言基础

1.1　能 力 要 求

（1）理解数据类型以及类型计算的思想，掌握关键字 typedef 的用法，理解并掌握 C++中的命名空间。

（2）掌握数据类型的基本运算和常用运算，理解左值和右值的概念。

（3）熟练应用 C++语言的各种类型的语句和控制结构。

（4）熟练应用 C++语言新增的内联函数、函数参数的默认值、函数重载、函数模板等。

（5）理解引用的概念，熟练掌握引用作为函数参数，理解函数返回引用的用法。

（6）掌握数组、指针与字符串。熟练应用 STL 常用算法、vector 容器、string 类型。

（7）掌握结构类型的定义和应用。理解链表结构，熟练应用 list 容器。

（8）理解模块化程序设计和结构化程序设计思想和方法，能解决比较简单的问题。

1.2　重点和难点

【重点】

（1）数据类型的基本运算（定义、赋值、初始化）。

（2）命名空间的定义及成员的访问。

（3）左值与右值的概念。常用运算（算术运算、赋值运算、增量/减量运算、关系运算、逻辑运算等）。

（4）语句与控制结构（顺序结构、选择结构和循环结构）。

（5）内联函数、函数参数的默认值、重载函数、函数模板。

（6）引用的语法。引用作为函数参数。

（7）STL 中计算数组区间的算法，vector、list 容器和 string 类型。

（8）结构类型的定义与应用。

【难点】

（1）数据类型的概念以及类型计算的思想。

（2）函数定义、函数调用的机制以及参数传递的方式、引用与函数。

（3）函数指针及其作为函数参数的用法。

（4）字符串处理及其运算。

（5）内存动态分配和释放。

1.3 知识点归纳

（1）分类（classification）是人们认识事物和概念的有效方法和工具。对具有共同属性的某些数据进行分类描述和计算，就得到数据类型（data type）。数据类型是一种可计算的模型，它描述了数据的内在属性，提供了数据运算的各种能力，并用唯一的符号来标识这种数据类型（数据类型标识符）。在定义每种数据类型时，应定义其结构特性和运算集合，并提供一个全局唯一的类型标识符（如 int、char 等）。类型的结构特性决定了该类型变量的存储方式和取值范围。运算集合（operation）决定了变量的运算能力，它包括所有类型都应该支持的基本运算（如定义、赋值和初始化），还包括常用运算（如算术运算、赋值运算、增量/减量运算、关系运算、逻辑运算、输入/输出等）。类型标识符是所有这些功能的体现，它所具有的功能一般应通过其实例（变量或对象）来体现。“定义类型、生成变量、实现计算”是类型计算的核心思想。

（2）学习 C/C++语言以及面向对象的思想和方法，就是定义并应用各种数据类型处理数据的过程。基本数据类型（包括整型、浮点型和 void 类型）是不需要定义的，它们在系统内置并得到天然支持，对它们的各种运算可以在系统中直接进行。复合数据类型（如数组、指针和 C 字符串）是基本数据类型的延拓，其运算也会得到系统各种形式的支持（内置功能，或者函数库）。但是更多的数据类型是需要用户自定义的，称为用户自定义类型（user defined types，UDT），包括枚举（enum）、联合（union）、结构（struct）和类（class），这些机制提供了对基本数据类型的不同封装形式和计算能力。定义类（和结构）是学习 C++和面向对象程序设计的中心工作。

（3）所有变量都有类型。辨别变量和类型、分析变量的类型是程序设计的基本能力。大多数变量的类型信息能够从其定义中直观地给出。对于一些隐式的类型（数组类型、指针类型、字符串类型、函数类型、引用类型），可以通过关键字 typedef 显式定义。

（4）定义（definition）、赋值（assignment）和初始化（initialization）是所有数据类型的基本运算能力。定义是为变量分配内存使得它存活的过程。赋值是对已定义变量所具有的数值进行修改的过程。初始化是在生成一个变量的同时为其设定初值的过程。它们由不同的函数过程所支持。

（5）C++提供流插入运算（运算符为左移运算符<<）和流提取运算（运算符为右移运算符>>）实现数据的输出和输入。运算符 sizeof 提供了分析变量所属类型的能力，它可以查看类型（或其变量）所占内存字节数。STL 函数调用表达式 numeric_limits<T>::max()和 numeric_limits<T>::min()分别返回基本数据类型 T 所能表示的最大值和最小值。

（6）命名空间是 C++语言解决标识符命名冲突的有力工具。通过关键字 namespace 可以定义命名空间。指令 using namespace NS 可以向程序中引入命名空间 NS 中所有的标识符。常用标识符 cout、cin、endl 定义在 C++标准命名空间 std 中。

（7）左值/右值是 C++表达式的一种属性。左值表达式可以存储数据、可被修改；右值表达式不能被修改。C++左值/右值有这些形式：①变量既可以用作左值，也可以用作右值。常量只能用作左值。②赋值运算表达式（包括复合赋值运算表达式）是左值。算术运算、关系运算和逻辑运算表达式是右值。最后一个子表达式是左值的逗号表达式是左值。两个备选子表达式都是左值的条件表达式是左值。③前增量/减量运算表达式是左值，后增量/减量运算表达式是右值。④所有正确返回引用的函数调用表达式是左值。非 const 版下标运算（实现为下标运算符[]或者函数调用运算符()）和转换运算符表达式是左值，const 版下标运算（实现为下标运算符[]或者函数调用运算符()）和转换运算符表达式是右值。⑤所有正确返回指针的函数调用表达式可用作左值。⑥正确返回*this、对应返回类型设为引用的类成员函数调用表达式是左值。

（8）STL 算术运算函数对象 plus<T>()、minus<T>()、multiplies<T>()、divides<T>()、modulus<T>()、negate<T>()分别实现类型 T 的算术加、减、乘、除、取余和一元取反运算。STL 关系运算函数对

此为试读，需要完整PDF请访问：www.ertongbook.com

象 greater<T>()、greater_equal<T>()、less<T>()、less_equal<T>()、equal_to<T>()、not_equal_to<T>()分别实现类型 T 的大于、大于等于、小于、小于等于、等于和不等于运算。注意，这 6 种运算基于<和==的等价实现形式。STL 逻辑运算函数对象 logical_not<T>()、logical_and<T>()、logical_or<T>()分别实现类型 T 的逻辑非、与、或运算。

（9）结构化程序设计的 3 种控制结构为顺序结构、选择结构和循环结构。顺序执行是所有程序流程的基本顺序。选择结构常用于“根据不同条件，执行不同动作”，C++提供 if 语句和 switch 语句实现选择判断。循环结构常用于“根据适当条件，重复执行多遍”，C++提供 while 语句、do...while 语句和 for 语句实现循环执行。转向语句 break、continue 和 return 可用于不同场合的程序流程跳转。

（10）函数原型、函数定义和函数调用是函数的 3 个重要概念。函数原型是对函数所有信息的声明，函数定义是函数功能实现的过程，函数调用是函数功能的体现。

（11）内联函数（关键字为 inline）、函数参数的默认值、函数重载、函数模板（关键字为 template）是 C++新增的 4 种函数机制。内联函数以文本替换的方式实现函数调用的快速执行。函数的默认参数提供了函数实参提供的灵活形式，它们应该在函数声明时从右向左设置。函数重载增强了程序可读性，使得同一函数可以适用于不同类型的数据。函数模板实现了程序代码的高度重用。

（12）引用是被引用变量的别名。所有对引用的操作都是对被引用变量的操作。

（13）函数参数传递的方式有 3 种：值传递、指针传递和引用传递。引用传递函数参数的优点在于：传递效率高（不生成实参的副本）、语法简单（与值传递语法相同）。在 3 种情况下宜选用引用传递函数参数：作为函数的输出参数，返回多个值；作为函数的输入参数（同时作为函数的输出参数），保留函数内部对参数的修改；作为输入参数，传递大对象（例如结构变量和类的对象）。

（14）函数返回数据的方式也有 3 种：值返回、指针返回和引用返回。引用返回数据的优点在于：返回效率高（不生成返回值的副本）。但是，引用返回（和指针返回）有很多限制，在 4 种情况下可以返回引用：返回对全局变量的引用；返回对局部静态变量的引用；返回对堆变量的引用；返回函数的引用参数或指针参数。

（15）数组具有很多应用上的优点：静态分配内存、连续存储、随机访问。对数组元素的访问可以用下标形式，更常见的是区间形式。左闭右开的区间形式是 STL 的表示形式。要正确设置数组作为函数参数时的函数形参：当一维数组用作函数参数时，需要把该数组的首地址和元素个数同时设置为函数的参数；当二维数组用作函数参数时，一般用指向数组的指针作为形参（对应二维数组类型），同时还要分别设置表示行数和列数的参数。

（16）实现数组常用计算的 STL 算法有：for_each 把某个操作（实现为函数）逐一施加到每个数组元素；copy 实现源区间数据到目标区间的复制；merge 实现两个有序区间的复制；find/find_if 在区间中查找某个元素（或符合某条件的元素）第一次出现的位置；binary_search 在有序区间中查找某个元素是否出现；max_element/min_element 分别计算区间值最大/最小元素的第一次出现位置；count/count_if 统计区间某个元素（或符合某条件的一些元素）出现的次数；fill/fill_n、generate/generate_n 可以以不同形式填充区间元素；sort 以非递减顺序对区间排序；reverse 逆转区间元素。

（17）vector 容器可以替代 C/C++数组，除了具有 C/C++数组的一些优点（如随机访问），它还具有动态分配内存、访问元素个数、灵活增删元素、相互赋值和复制等优点。熟练应用 vector 容器，需要掌握：容器的构造（给定长度构造和区间形式构造）和赋值（=和 assign）；访问容器大小（size 和 empty）和存取元素（下标形式和迭代器形式）；添加元素（push_back 和 insert）和删除元素（pop_back、erase 和 clear）。

（18）指针是 C/C++重要的数据类型，要熟练掌握指针的定义、运算（取地址和去引用、算术运算和关系运算）。正确设置指针类型的函数形参，并能传入对应的实参。

（19）以函数作为参数能够表示动作执行的方式和策略，这需要用到函数指针（以及函数指针类型）的概念。能够正确设置函数指针类型的形参和实参，并能结合函数模板设置函数类型的形参和实参。

（20）运算符 new/delete 是实现 C++动态内存分配和释放的有力工具。能够掌握堆变量和数组的动态生成和释放。

（21）C 字符串（以'\0'作为字符串结束符）可以字符数组形式存储，也可通过字符指针访问。C 头文件<string.h>（或 C++标准头文件<cstring>）提供了大量 C 字符串处理的函数，例如函数 strlen()计算字符串的长度，函数 strcpy()实现字符串复制，函数 strcmp()比较字符串内容，函数 strcat()实现字符串连接，函数 strtok()按记号分割字符串。

（22）STL 字符串类 string（C++标准头文件<string>）提供更为灵活的字符串处理方式。与 C 字符串不同的是，string 不以'\0'作为字符串结束符，它的长度可以通过方法 size 或者 length 获取。与 C 字符串相同的是，它们都可以用下标运算符访问字符元素。

（23）熟练应用 STL 的 string 类型，需要掌握：字符串的构造（空对象、以 C 字符串构造）、赋值（=或者 assign）、输入（全局函数 getline()）、输出（<<）；访问元素个数（size）和存取元素（下标形式、迭代器形式或者 at）；插入字符（insert）、删除字符（erase）和追加元素（append）；截取字符串（substr）和字符串连接（+、+=）；字符或子字符串查找（find、rfind、find_first_of、find_last_of、find_first_not_of、find_last_not_of）、字符串比较（compare 或者关系运算符）、转换为 C 字符串（c_str）。string::npos 通常表示字符串中一个不存在的值，例如成员函数 find()的返回值。

（24）结构类型（struct）能够把某事物的多个属性封装为一个整体，这些属性表示为结构类型的成员。定义结构类型之后，可以定义结构变量、结构指针和结构引用，它们对成员的访问分别用圆点（.）和箭头（->）成员访问运算符。结构类型作为函数参数，一般以引用传递或 const 引用传递。

（25）链表是实现数据动态存储的数据结构。STL 容器 list 提供了链表的功能。如果需要经常在容器的中间或者两端进行元素插入和删除操作，就应该选用 list 容器。熟练应用 list 容器，需要掌握：构造（默认构造空链表、以区间形式构造）和赋值（=和 assign）；访问容器大小（size 和 empty）和存取元素（迭代器形式、front 和 back）；插入元素（insert、push_front/push_back）和删除元素（clear、remove、erase、pop_front/pop_back）；链表合并（splice、merge）、逆转（reverse）和排序（sort）。

测 试 题

一、判断题

1. 函数原型 int f(int = 2, int, double = 3);是错误的。（ ）
2. 在 switch 语句中，default 关键字只能放在所有 case 标号的后面。（ ）
3. C++程序中，通常使用 const 来定义符号常量，定义时必须指出类型。（ ）
4. 增量表达式是左值表达式。（ ）
5. 在定义语句“int a(5), &b = a, *p = &a;”中，b 的值和 p 的值是相等的。（ ）
6. 在 for 循环中定义的循环变量的作用域是该循环的循环体内。（ ）
7. 函数可以设置默认的参数值，默认参数值必须设置在函数定义时的形参上。（ ）
8. 若有定义“int k = 0;”，则执行语句“do ++k; while (k * k < 20);”后，k 的值为 4。（ ）
9. 根据定义 int n = 1, &r = n;，则表达式++r, n--的值为 1。（ ）
10. 若有定义“typedef char* PCHAR;”，则可用语句“PCHAR p;”定义字符变量 p。（ ）

二、单项选择

1. 有如下程序段：

```
int i = 1;
while (1) {
    ++i;
    if(i == 10)  break;
    if(i % 2 == 0)  cout << '#';
}
```

执行这个程序段输出字符#的个数是（　　）。

A. 10　　B. 3　　C. 4　　D. 5

2. 下列函数参数默认值定义错误的是（　　）。

A. f(int x, int y = 0);

B. f(int x = 100);

C. f(int x = 0, int y);

D. f(int x = g()); //（假定函数 g()已经定义）

3. 现声明两个函数："void f(int, char = 'a');"，"void f(int);"，则（　　）。

A. 它们不能在同一程序中定义　　B. 它们可以在同一程序中定义并可以重载

C. 它们可以在同一程序中定义，但不可以重载　　D. 以上说法均不正确

4. 已知函数 f()的原型是 void f(int* x, int& y);，变量 x，y 的类型都是 int，下列调用语句中，正确的是（　　）。

A. f(x, y);　　B. f(x, &y);　　C. f(&x, y);　　D. f(&x, &y);

5. 下列对字符数组进行初始化的语句正确的是（　　）。

A. char str[] = "Hello";

B. char str[][] = {'H', 'e', 'l', 'l', 'o'};

C. char str[5] = "Hello";

D. char str[2][5] = {"Hello", "World"};

6. 已知 int a[3][4];下列表示中（　　）是错误的。

A. *(a+1)[3]　　B. *(*a+3)　　C. *(*(a+1))　　D. *(&a[0][0]+2)

7. 下列枚举类型的定义中，包含枚举值 3 的是（　　）。

A. enum test {RED, YELLOW, BLUE, BLACK};

B. enum test {RED, YELLOW=4, BLUE, BLACK};

C. enum test {RED=-1, YELLOW,BLUE, BLACK};

D. enum test {RED, YELLOW=6, BLUE, BLACK};

8. 下列选择重载函数的不同实现的判断条件中，错误的是（　　）。

A. 参数类型不同　B.参数个数不同　C. 参数顺序不同　D.函数返回值不同

9. 下列说明中 const char * ptr; ptr 应该是（　　）。

A. 指向字符的指针　　B. 指向字符的常量指针

C. 指向字符串常量的指针　　D. 指向字符串的常量指针

10. 以下程序的输出结果是（　　）。

```
int f() {
    static int i = 0;
    int s = 1;
    s += i;
    ++i;
    return s;
```

```
}
int main() {
    int i, a = 0;
    for(i = 0; i < 5; ++i)
        a += f();
    cout << a;
}
```

A. 20　　B. 24　　C. 25　　D. 15

11. 下面语句中函数原型正确的是（　　）。

A. double Function(void f);　　B. void Function (double);

C. double Function(f);　　D. void double(double f);

12. 下列定义语句中，错误的是（　　）。

A. int pa const*;　　B. char* pb[10];　　C. char (*pc) [10];　　D. int (*pd) ();

13. 定义函数如下：

```
int f(int a, int b = 1, int c = 2) {
    return a + b + c;
}
```

下列计算结果错误的是（　　）。

A. 表达式 f(1)的值为 4。　　B. 表达式 f(1, f(1))的值为 7。

C. 表达式 f(1, f(1), f(1))的值为 9。　　D. 表达式 f(f(1), f(1))的值为 11。

14. 有如下程序：

```
int main() {
    int s;
    for (int i = 0; i < 6; i += 3) {
        s = i;
        for (int j = i; j < 6; ++j)
            s += j;
    }
    cout << s << endl;
}
```

运行时的输出结果是（　　）。

A. 3　　B. 10　　C. 12　　D. 15

15. 现有函数原型 double Function (double, double, double);，则下列不能同时定义的有（　　）。

A. char Function (int, int);　　B. double Function (int, int, doubl);

C. int Function (int, char*);　　D. int Function (double, double, double);

16. 有如下定义：

```
int a[5] = {1, 3, 5, 7, 9}, *p = a;
```

下列表达式中不能得到数值 5 的是（　　）。

A. a[2]　　B. a[3]　　C. *(p+2)　　D. *p+4

17. 有如下程序段：

```
int x = 0, y = 1;
int& r = x;          //①
r = y;               //②
int* p = &x;         //③
*p = &r ;            //④
```

其中会产生编译错误的语句是（　　）。

A. ④　　B. ③　　C. ②　　D. ①

18. 有如下程序：

```
void f1(int& x, int& y) {
    int z = x; x = y; y = z;
}
void f2(int x, int y) {
    int z = x; x = y; y = z;
}
    int main() {
    int x = 10, y = 26;
    f1(x, y);
    f2(x, y);
cout << y << endl;
}
```

运行时的输出结果是（　　）。

A. 10　　B. 16　　C. 26　　D. 36

19. 在下列关于内联函数的描述中，（　　）是错误的。

A. 内联函数中可以包括各种语句

B. 对内联函数不可以进行异常接口声明

C. 内联函数主要解决程序的运行效率问题

D. 内联函数的定义必须出现在内联函数第一次被调用之前

20. 有以下定义语句：

```
struct student {
    int age;
    char num[20];
};
struct student stu[3] = {{20, "201201"}, {21,"201202"}, {19,"201203"}};
struct student *p = stu;
```

以下引用结构体变量成员的表达式错误的是（　　）。

A. (p++)->num　　B. p->num　　C. (*p).num　　D. stu[3].age

三、程序分析

1. 若下列程序运行时输出结果为：

```
1, A, 4.5
2, B, 3.5
```

请将程序补充完整。

```
void f(int, char, double_____________);
int main() {
    f(1, 'A', 10.1);
    f(2, 'B');
}
void f(int a, char b, double c) {
    cout << a << ", " << b << ", " << c << endl;
}
```

2. 下列函数的功能是判断字符串 str 是否回文（正向输出和逆向输出的结果相同），若是则返回 true，否则返回 false。请在横线处填上适当内容，实现该函数。

```
bool f(char* s) {
    int i = 0, j = 0;
    while (s[j]) _____________;
```

```
    for (j--; i < j && s[i] == s[j]; i++, j--)
        ;
    return _____;
}
```

3. 说明下列程序中的错误，并改正。

```
int& f(int a[], int idx) {
    int n = a[idx];
    return n;
}
```

4. 分析程序，写出运行结果。

```
int fun(char *, char *);
int main() {
    char *p1, *p2;
    p1 = "abcfgy";
    p2 = "abcdhu";
    int n = fun(p1, p2);
    cout << n << endl;
}
int fun(char *s1, char *s2) {
    while(*s1 && *s2 && *s1++ == *s2++);
    s1--;
    s2--;
    return *s1 - *s2;
}
```

5. 分析程序，写出运行结果。

```
void f(int a[], int i, int j) {
    int t;
    if(i < j) {
        t = a[i];
        a[i] = a[j];
        a[j] = t;
        f(a, i + 1, j - 1);
    }
}
int main() {
    int i, aa[5] = {1,2,3,4,5};
    f(aa, 0, 4);
    for(i = 0; i < 5; i++)
        cout << aa[i];
    cout << "\n";
}
```

6. 分析程序，写出运行结果。

```
double arr[5];
double& select(int num) {
    int n = num % 5;
    return arr[n];
}
int main() {
    for(int i = 11; i <= 15; ++i) {
        select(i) = i;
```

此为试读，需要完整PDF请访问：www.ertongbook.com