高等院校“十二五”核心课程辅导丛书

C++ 答疑解惑与典型题解

C++
DAYIJIEHUO YU DIANXINGTIJIE

汪名杰 尹 静 郝 立 编著

北京邮电大学出版社
www.buptpress.com

高等院校“十二五”核心课程辅导丛书

C++答疑解惑与典型题解

汪名杰　尹静　郝立　编著

北京邮电大学出版社
·北京·

内 容 简 介

本书深入浅出、系统全面地介绍了最新各大高校的C++练习题及考研题。全书共分22章，内容包括C++的基本概念，面向对象程序设计、函数、数组、指针、引用、结构、类、拷贝函数等。

本书以知识结构图—常见疑惑解答—实践解题编程—考研真题讲解为主线组织编写，每一章的题型归纳都进行了详细分析评注，以便于帮助读者掌握本章的重点及迅速回忆本章的内容。本书结构清晰、易教易学、实例丰富、学以致用、注重能力，对易混淆和历年考题中较为关注的内容进行了重点提示和讲解。

本书既可以作为C++编程学习的参考书，也可以作为复习考研的练习册，更可以作为各类培训班的培训教程。此外，本书也非常适于教师的C++教学以及各种编程自学人员参考阅读。

图书在版编目(CIP)数据

C++答疑解惑与典型题解/汪名杰，尹静，郝立编著. --北京：北京邮电大学出版社，2010.9

ISBN 978-7-5635-2290-3

Ⅰ.①C… Ⅱ.①汪…②尹…③郝… 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(2010)第172519号

书　　名：C++答疑解惑与典型题解

主　　编：汪名杰　尹静　郝立

责任编辑：满志文

出版发行：北京邮电大学出版社

社　　址：北京市海淀区西土城路10号(邮编：100876)

发 行 部：电话：010-62282185　传真：010-62283578

E-mail：publish@bupt.edu.cn

经　　销：各地新华书店

印　　刷：北京源海印刷有限责任公司

开　　本：787 mm×1 092 mm　1/16

印　　张：21.25

字　　数：529千字

版　　次：2010年9月第1版　2010年9月第1次印刷

ISBN 978-7-5635-2290-3　　定　价：35.00元

前　言

为适应高等院校人才的考研需求，本书本着厚基础、重能力、求创新的总体思想，着眼于国家发展和培养造就综合能力人才的需要，着力提高大学生的学习能力、实践能力和创新能力。

1. 关于C++程序设计

C++是计算机程序设计的重要理论和技术基础，它是一种混合型的面向对象程序设计语言。随着时代的发展，它现在不仅仅是计算机学科的重要课程，而且进一步成为一些其他理工科学习的必备技术。它既具有独特的面向对象特征，又具有传统C语言的向后兼容性，具备结构化程序设计特征。为编程者能编出简单高效的程序打下一个良好的基础，特别为学习掌握Visual C++、Java等软件开发工具提供了坚实的理论基础。

2. 本书阅读指南

本书基于C++程序设计的基础，针对C++知识点的常见的问题进行了讲解，同时分析了近几年的考研题目，并给出了翔实的参考答案，读者可以充分地了解各个学校考研题目的难度，查缺补漏，有针对性地提高自己的水平。本书共分22章。

第1章主要讲解C++的基本概念和入门。

第2章主要讲解C++的基本数据类型和自定义数据类型。

第3章主要讲解C++的表达式和编程语句。

第4章主要讲解C++的过程化语句。

第5章主要讲解C++的函数，函数是C++的核心所在，本章对函数的定义用法等进行了详细讲解。

第6章主要讲解C++的程序结构。

第7章主要讲解C++的另一个要点——数组，对数组的用法等进行详细讲述。

第8章主要讲解C++的重点知识——指针，通过例题进行了指针用法等的讲解说明。

第9章主要讲解C++中引用的相关知识。

第10～第18章着重对C++的核心知识进行讲解，包括结构、类、构造函数、面向对象程序设计、堆与拷贝构造函数、静态成员与友元、继承与派生、多重继承、多态性与虚函数等。

第19～第21章主要是对I/O流、模板以及C++工具进行讲解。

第22章提供了两套模拟题，为读者提供一个自我分析解决问题的过程。

本书的重点在中间的5、7、8等几章。

3. 本书特色与优点

(1) 结构清晰，知识完整。内容翔实、系统性强，依据高校教学大纲组织内容，同时覆盖最新版本的所有知识点，并将实际经验融入基本理论之中。

(2) 内容翔实，解答完整。本书涵盖近几年各大高校的大量题目，示例众多，步骤明确，

讲解细致，读者不但可以利用题海战术完善自己的弱项，更可以有针对性地了解某些重点院校的近年考研题目及解题思路。

（3）学以致用，注重能力。一些例题后面有与其相联系的知识点详解，使读者在解答问题的同时，对基础理论得到更深刻的理解。

（4）重点突出，实用性强。

4. 本书读者定位

本书既可以作为复习考研的练习册，也可以作为数据结构学习的参考书，更可以作为各类培训班的培训教程。此外，本书也非常适于教师的数据结构教学以及各种编程自学人员参考阅读。

本书由汪名杰、尹静、郝立编著，全书框架结构由何光明、吴婷拟定。另外，感谢王珊珊、陈智、陈海燕、吴涛涛、李海、张凌云、陈芳、李勇智、许娟、史春联等同志的关心与帮助。

由于作者水平有限，书中难免存在不当之处，恳请广大读者批评指正。如遇到疑难问题，可通过以下方式与我们联系：bjbaba@263.net。

编　者

目　录

第 1 章　C++基本概念和入门 …… 1

1.1　答疑解惑 …… 2

1.1.1　C++程序是如何构成的？ …… 2

1.1.2　在 C++程序中，主函数、标准库函数和用户自定义函数之间有何区别？ …… 2

1.1.3　在 C++程序中，函数是如何构成的？ …… 2

1.1.4　如何将 C++源程序生成可执行的程序？ …… 2

1.1.5　为什么要在程序中使用注释，C++中有何注释方法？ …… 3

1.1.6　C 语言和 C++有什么关系？ …… 3

1.1.7　为什么编译系统要提供标准库函数，如何使用标准库函数？ …… 3

1.1.8　C++程序的构成和书写形式是什么？ …… 4

1.1.9　如何理解程序设计的目标在正确的前提下，其重要性排列次序依次为：可读、可维护、可移植和高效率？ … 5

1.1.10　什么是数值溢出？整数溢出与浮点数溢出有何不同？ …… 5

1.2　典型题解 …… 6

题型 1　基础知识 …… 6

考研真题汇总 …… 7

第 2 章　基本数据类型与自定义数据类型 …… 10

2.1　答疑解惑 …… 10

2.1.1　基本数据类型和非基本数据类型（自定义数据类型）有何区别？ …… 10

2.1.2　为什么在定义变量时要指定变量的类型，如何确定一个变量的类型？ …… 11

2.1.3　C++中有哪几种指定常量的方法，为何要定义常变量？ …… 11

2.1.4　C++中字符型常量与字符串常量的主要区别是什么？ …… 12

2.1.5　什么是匈牙利命名法，如何使用它？ …… 12

2.1.6　如何确定一个类型变量所占的内存字节数，某个基本数据类型的变量占用的内存字节数都是固定的吗？ … 13

2.1.7　保留字与标准标识符之间有何区别？ …… 13

2.1.8　“0”、‘0’、‘\0’和“\0”之间有何区别？ …… 13

2.2　典型题解 …… 14

题型 1　基础知识 …… 14

题型 2　程序理解 …… 18

题型 3　编程实践 …… 19

考研真题汇总 …… 23

第 3 章　表达式和编程语句 …… 26

3.1　答疑解惑 …… 26

3.1.1　什么是 C++的运算符、表达式和语句？ …… 26

3.1.2　什么是运算符的优先级和结合性？ …… 27

3.1.3　C++中 bool 类型值有哪些表示方法？关于布尔的运算说明有哪些？请举例说明 …… 28

3.1.4　什么是短路表达式？ …… 28

3.1.5　什么是左值和右值？ …… 29

3.1.6　设有“int a;”，则 if(a=3)与 if(a==3)有何区别？ …… 30

3.1.7　浮点变量与数字常量之间使用“==”和“!=”进行比较可靠吗？ …… 30

3.1.8　什么是操作数的求值顺序，它有什么副作用？ …… 30

3.1.9　在 32 位机器中，sizeof(int)为 4，为什么 cout≪747483647 * 7/7≪endl；显示的结果不是 747483647？ …… 31

3.1.10 switch 和 if...else...可随意替换么？ …… 31
3.2 典型题解 …… 32
题型 1 基础知识 …… 32
题型 2 表达式理解 …… 34
题型 3 编程实践 …… 36
考研真题汇总 …… 38

第 4 章 过程化语句 …… 42

4.1 答疑解惑 …… 42
4.1.1 switch 语句中必须使用 default 分支吗？ …… 42
4.1.2 while 循环与 do...while 循环有什么不同？ …… 42
4.1.3 break 语句和 continue 语句的区别是什么？ …… 43
4.1.4 各种循环语句的总体结构是怎样的？ …… 43
4.2 典型题解 …… 44
题型 1 基础知识 …… 44
题型 2 程序理解 …… 45
题型 3 编程实践 …… 49
考研真题汇总 …… 54

第 5 章 函数 …… 55

5.1 答疑解惑 …… 55
5.1.1 为什么要使用函数？ …… 55
5.1.2 函数的设计规则是什么？ …… 56
5.1.3 C++中函数可以分为哪几类？ …… 56
5.1.4 为什么需要函数原型？ …… 56
5.1.5 如何让主调用函数“找到”被调用函数？ …… 56
5.1.6 如何在全局变量的定义位置之前使用它？ …… 57
5.1.7 递归可以解决什么样的问题？ …… 57
5.1.8 使用内联函数有什么好处，对内联函数有什么要求？ …… 57
5.1.9 使用重载函数有什么好处，系统如何区分被重载的函数？ …… 57
5.2 典型题解 …… 58
题型 1 基础知识 …… 58
题型 2 程序理解 …… 59
题型 3 编程实践 …… 62
考研真题汇总 …… 65

第 6 章 程序结构 …… 72

6.1 答疑解惑 …… 72
6.1.1 一个大型 C++源程序的结构是怎样的？系统如何生成可执行文件？ …… 72
6.1.2 有变量定义“register float f;”该变量定义有何不恰当的地方？ …… 73
6.1.3 什么是预处理？ …… 73
6.1.4 #include <文件名>和#include “文件名”有区别吗？ …… 73
6.1.5 头文件中一般包含什么信息？ …… 73
6.1.6 if...else...和#if...#else...有何区别？ …… 74
6.1.7 文件包含命令可以嵌套吗？ …… 74
6.1.8 如何避免多次包含同一个头文件？ …… 75
6.1.9 什么是 void 函数，它怎么使用？ …… 75
6.1.10 头文件的结构是什么？请举例说明 …… 75
6.2 典型题解 …… 76
题型 1 基础知识 …… 76
题型 2 程序理解 …… 78
题型 3 编程实践 …… 79
考研真题汇总 …… 79

第 7 章 数组 …… 89

7.1 答疑解惑 …… 89
7.1.1 什么是数组，为什么需要数组？ …… 89
7.1.2 数组在内存中如何存放，一个数组究竟要占用多少内存？ …… 90
7.1.3 若有数组定义语句“int a[5];”，则 a 代表什么？&a[1]-&a[0]=？ …… 90
7.1.4 为什么数组名必须是常量？ …… 91
7.1.5 传递数组时是将整个数组的内容都传递给函数吗？ …… 91
7.1.6 若有定义"char s1[]="well";char s2[]={'w','e','l','l'};",则 s1 和 s2 相同吗？ …… 92
7.1.7 数组的下标越界时，编译器并不指示错误，是不是下标越界与否无关紧要？ …… 92

7.1.8 数组名和指针是不是同一回事？ …… 92
7.2 典型题解 …… 94
题型1 基础知识 …… 94
题型2 程序理解 …… 97
题型3 编程实践 …… 98
考研真题汇总 …… 102

第8章 指针 …… 121

8.1 答疑解惑 …… 121
8.1.1 内存单元的地址和内存单元的内容相同吗？ …… 121
8.1.2 基类型不同的指针所占用的内存字节数相同吗？如何取得一个变量的内存地址？ …… 122
8.1.3 指针变量的地址就是指针所存放的地址吗？ …… 122
8.1.4 为什么要对指针变量进行初始化？对指针进行初始化有哪些方法？ …… 122
8.1.5 指针＋整数＝？指针－指针＝？ … 123
8.1.6 两个指针可以相比较吗？ …… 124
8.1.7 为什么需要动态内存分配？C＋＋程序中动态分配的内存会被自动释放吗？ …… 124
8.1.8 如何通过指针以及数组名来引用一维数组的元素？ …… 125
8.1.9 如何通过指针以及数组名来引用二维数组的元素？ …… 125
8.1.10 使用指针变量操作字符串与字符数组操作字符串的区别是什么？ …… 127
8.1.11 指针函数和函数指针含义相同吗？ …… 127
8.1.12 指针数组和数组指针是否相同？ … 129
8.1.13 常量指针，指针常量和指向常量的指针常量有何区别？ …… 129
8.2 典型题解 …… 130
题型1 基础知识 …… 130
题型2 程序理解 …… 133
题型3 编程实践 …… 137
考研真题汇总 …… 141

第9章 引用 …… 149

9.1 答疑解惑 …… 149
9.1.1 什么是引用？ …… 149
9.1.2 “任何东西都可以定义一个引用”这句话对吗？ …… 150
9.1.3 函数传递参数有哪些方式？ …… 150
9.1.4 引用和指针之间有何区别？ …… 152
9.1.5 下面两个引用运算符 & 的用法是否相同？ …… 153
9.1.6 “引用没有对应的内存，因此不能对引用运用 & 运算符来取地址”这句话对否？ …… 153
9.2 典型题解 …… 153
题型1 基础知识 …… 153
题型2 程序理解 …… 154
题型3 编程实践 …… 155
考研真题汇总 …… 157

第10章 结构 …… 162

10.1 答疑解惑 …… 162
10.1.1 结构体与结构体变量有何区别？ …… 162
10.1.2 在C＋＋中结构体与类有何区别？ …… 163
10.1.3 为什么在 Turbo C＋＋3.0 中编译以下程序会出现“Cannot convert ‘B’ to ‘A’”的错误提示？ …… 163
10.1.4 要访问结构体变量的成员，有哪些方法？ …… 163
10.1.5 在32位机器中，若有定义：struct STU {char c1;int i;char c2} s;则sizeof(s)＝＝6吗？ …… 164
10.1.6 结构体和联合体(共用体)的异同点是什么？ …… 164
10.2 典型题解 …… 165
题型1 基础知识 …… 165
题型2 程序理解 …… 167
题型3 编程实践 …… 170
考研真题汇总 …… 173

第11章 类 …… 183

11.1 答疑解惑 …… 183

11.1.1 结构体类型和类有什么区别？ …… 183
11.1.2 一个类的接口和实现有何区别？以及构造函数和析构函数的区别有哪些？ …… 184
11.1.3 什么是对象，如何定义和使用对象？ …… 184
11.1.4 类的成员有哪些访问权限？ …… 184
11.1.5 将类的成员设置成 private 访问权限可以防止一切非法的访问吗？ … 184
11.1.6 类里面的常量都有效吗？ …… 185
11.2 典型题解 …… 186
题型 1 基础知识 …… 186
题型 2 程序理解 …… 187
题型 3 编程实践 …… 192
考研真题汇总 …… 195

第 12 章 构造函数 …… 204

12.1 答疑解惑 …… 204
12.1.1 构造函数的作用是什么？它有什么特点？ …… 204
12.1.2 析构函数有什么特点？它的作用是什么？ …… 205
12.1.3 类的默认构造函数有哪些？ …… 206
12.1.4 以下代码的输出结果是什么？ … 206
12.1.5 为什么下面的代码不能通过编译？ …… 207
12.2 典型题解 …… 208
题型 1 基础知识 …… 208
题型 2 程序理解 …… 208
题型 3 编程实践 …… 214
考研真题汇总 …… 217

第 13 章 面向对象程序设计 …… 219

13.1 答疑解惑 …… 219
13.1.1 什么是结构化程序设计，它有何缺点？ …… 219
13.1.2 析构函数有什么特点？它的作用是什么？ …… 220
13.1.3 为什么使用面向对象编程比面向结构编程更能提高程序员的工作效率？ …… 220
13.1.4 为什么使用面向对象编程比面向结构编程更能提高程序员的工作效率？ …… 221
13.2 典型题解 …… 224
题型 1 基础知识 …… 224
题型 2 程序理解 …… 224
题型 3 编程实践 …… 225
考研真题汇总 …… 229

第 14 章 堆与拷贝构造函数 …… 233

14.1 答疑解惑 …… 233
14.1.1 什么是堆？ …… 233
14.1.2 为何要用 new 和 delete 来从堆中动态分配对象，而不使用 malloc 和 free？ …… 234
14.1.3 如何正确使用 malloc 和 free？ …… 234
14.1.4 若 Student 是已经定义的类，为什么构造函数 Student::Student (Student)是非法的？ …… 236
14.1.5 构造函数与析构函的数的联系是什么？ …… 237
14.1.6 构造与析构的次序是什么？ …… 237
14.2 典型题解 …… 237
题型 1 基础知识 …… 237
题型 2 程序理解 …… 238
题型 3 编程实践 …… 242
考研真题汇总 …… 243

第 15 章 静态成员与友元 …… 245

15.1 答疑解惑 …… 245
15.1.1 为什么下面的代码在编译时通不过？ …… 245
15.1.2 访问静态成员有哪些方式？ …… 246
15.1.3 为什么下面的代码编译时，通不过编译，提示：void _cdecl Sample::print(void)' : function cannot access 'Sample::x',print 是 Sample 的友元函数，应该可以访问类的私有成员啊？ …… 246
15.1.4 源文件中定义函数或函数中定义静态变量时使用的 static 和在类中说明静态成员时使用的 static 有何区别？ …… 247
15.2 典型题解 …… 247

题型 1 基础知识 …… 247
题型 2 程序理解 …… 247
题型 3 编程实践 …… 252
考研真题汇总 …… 253

第 16 章 继承与派生 …… 254

16.1 答疑解惑 …… 254
16.1.1 什么是继承，继承有何作用？ … 254
16.1.2 组合和继承的区别是什么？在一个继承的层次结构中，默认的构造函数和析构函数是如何区分的？ … 255
16.1.3 派生类的对象是如何被构造和析构的？ …… 255
16.1.4 继承和组合创建新类有何区别？ …… 255
16.1.5 什么是多态，如何理解动态联编和静态联编？ …… 256
16.1.6 为什么下面的代码通不过编译，提示：cannot convert parameter 1 from 'const int' to 'int *'，derive 类中不是继承 base 类中的 int add(int, int)；了吗？ …… 257
16.1.7 为什么下面的代码没有实现我们所需要的多态？ …… 257
16.2 典型题解 …… 258
题型 1 程序理解 …… 258
题型 2 编程实践 …… 267
考研真题汇总 …… 270

第 17 章 多重继承 …… 271

17.1 答疑解惑 …… 271
17.1.1 不同的继承方式对从基类中继承来的成员的访问权限有何影响？ … 271
17.1.2 下面的代码，为何函数调用 disp(&d2)；会出错？ …… 272
17.1.3 为什么下面的代码不能通过编译？提示：'base::base'：no appropriate defaultconstructor available …… 273
17.2 典型题解 …… 273
题型 1 基础知识 …… 273
题型 2 程序理解 …… 274
题型 3 编程实践 …… 278

第 18 章 多态性与虚函数 …… 281

18.1 答疑解惑 …… 281
18.1.1 多态性的概念是什么？ …… 281
18.1.2 在 C++中关键字 operator 有何作用？ …… 281
18.1.3 运算符作为成员函数重载和作为友元函数重载有何区别？ …… 282
18.1.4 为什么流运算符“<<”和“>>”重载时必须作为友元函数出现？ …… 282
18.1.5 前置的自增运算符和后置的自增运算符在重载时有何区别？ …… 283
18.1.6 以下程序运行结果是什么？ …… 285
18.1.7 在 C++中虚函数有何作用？在什么情况下应该声明虚函数？ …… 285
18.2 典型题解 …… 286
题型 1 基础知识 …… 286
题型 2 程序理解 …… 286
题型 3 编程实践 …… 292
考研真题汇总 …… 294

第 19 章 I/O 流 …… 296

19.1 答疑解惑 …… 296
19.1.1 在 C++中如何使用流类？ …… 296
19.1.2 输入/输出的含义是什么？在 C++中输入/输出包括哪些内容？ …… 296
19.1.3 printf( )与 scanf( )函数有什么缺陷？ …… 297
19.1.4 C++如何简化字符串、外部文件和内部文件的处理？ …… 297
19.1.5 有序访问与直接访问的区别是什么？ …… 297
19.1.6 I/O 标准流类、文件流类和串流类的头文件和关联的关系如何？ …… 297
19.2 典型题解 …… 298
题型 1 基础知识 …… 298
题型 2 程序理解 …… 298
题型 3 编程实践 …… 299
考研真题汇总 …… 300

第 20 章 模板 …… 302

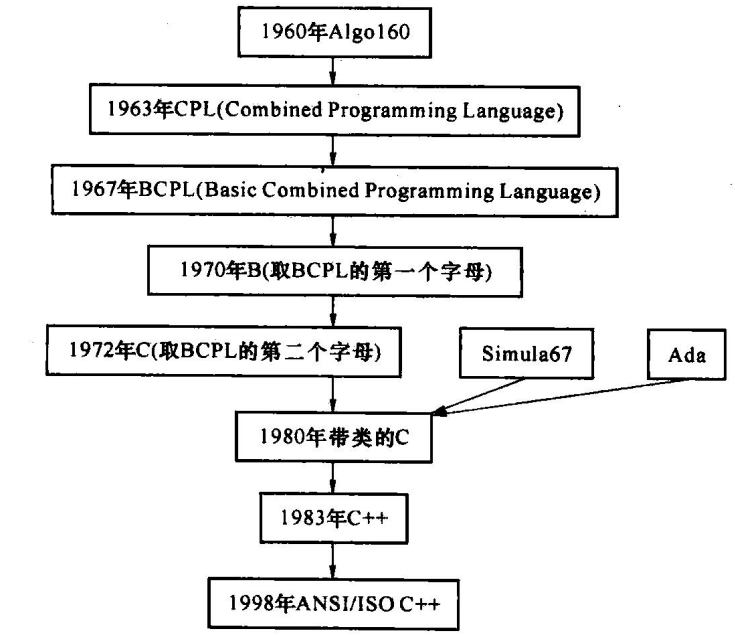
20.1 答疑解惑 …… 302

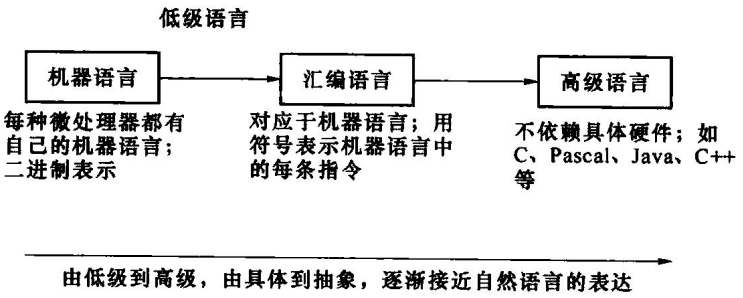
20.1.1 如何区分以下概念：函数模板与模板函数、类模板与模板类？ … 302
20.1.2 与宏定义相比，采用函数模板有什么好处？ … 302
20.1.3 以下程序的运行结果是什么？ … 303
20.2 典型题解 … 303
题型 1 基础知识 … 303
题型 2 程序理解 … 303
题型 3 编程实践 … 304
第 21 章 C++工具 … 305
21.1 答疑解惑 … 305
21.1.1 在 C++中为什么要引入异常处理？ … 305
21.1.2 C++中异常是如何被捕获的？ … 305
21.1.3 异常处理族系的构成有哪两种办法？ … 306
21.1.4 C++中为什么需要命名空间？ … 308
21.1.5 命名空间的概念是什么？如何使用？ … 308
21.2 典型题解 … 309
题型 1 基础知识 … 309
题型 2 程序理解 … 310
题型 3 编程实践 … 311
第 22 章 课程测试及考研仿真题 … 312
22.1 课程测试 … 312
22.2 课程测试参考答案 … 319
22.3 考研仿真题 … 320
22.4 考研仿真题参考答案 … 326
参考文献 … 329

第1章

C++基础概念和入门

知识结构图





1.1 答疑解惑

1.1.1 C++程序是如何构成的?

C++程序是由函数构成的,函数是构成C++程序的基本单位。一个可执行C++程序至少包含一个主函数,除此之外还可以包含其他的函数。C++程序有3种类型的函数:

(1) 主函数,即main( )函数;

(2) 标准库函数,如strcmp,strcpy等;

(3) 用户自定义函数,如自己定义的max等。

1.1.2 在C++程序中,主函数、标准库函数和用户自定义函数之间有何区别?

主函数是一个用户定义函数,但是它和其他的用户定义函数的区别为:

(1) 主函数名必须是main,不能够由用户自己随便设置;

(2) 每个C++程序都有且只有一个主函数,而其他的用户自定义函数可以有多个;

(3) 每个可执行C++程序必须从主函数开始执行。

标准库函数和用户定义函数之间的区别为:

(1) 用户自定义函数是由程序员自己定义和实现的;

(2) 标准库函数是由随编译器软件一起由软件厂商提供的,如strcmp,strcpy等。

主函数、标准库函数和用户自定义函数在本质上是没有区别的,它们都是函数,定义和调用的方式都是相同的,只是编写函数的人不同。另外注意一点,main函数是由系统调用的,程序员在源程序中不要再调用该函数。

1.1.3 在C++程序中,函数是如何构成的?

在C++程序中一个函数由两个部分组成:

(1) 函数首部:函数首部包括函数返回值类型、函数名,函数参数列表。

(2) 函数体:在函数首部下面用{ }括起来的部分。其内包含若干条C++语句,用来实现函数的功能。

如:

```
double min(double x, double y)
  //函数首部:返回 double 类型值,函数名为 min,两个参数 x,y 都是 double 类型
  { //函数体开始
        if (x < y)
        return x;
  else
        return y;
  } //函数体结束
```

1.1.4 如何将C++源程序生成可执行的程序?

C++源程序的扩展名一般为".cpp",将C++源程序转换成可执行的程序要经过两步:

(1)编译:将以文本形式存储的扩展名为".cpp"的源程序用适当的编译器(一个可执行程序)编译成二进制形式存储的目标文件,扩展名为".obj"。

(2)连接:将多个目标文件以及使用到的标准库用连接器(一个可执行的程序)连接在一起构成可执行的文件,一般扩展名为".exe"。

1.1.5 为什么要在程序中使用注释,C++中有何注释方法?

(1) 实时恰当的注释能够提高程序的可读性。在团队开发中,养成良好的编程风格是十分必要的,这可以让协作者、后继者和自己在以后一目了然,在短时间内看清楚程序的结构,理解程序的设计思路;良好的编程风格应该是初学时逐步形成,这有利于以后形成更好的编程思维,从而提高程序开发的效率。通过注释一般只能在局部很快了解程序的结构和思路,要了解一个大程序的整体结构,则完整的设计文档是必不可少的。

(2) 注释对于程序的调试具有非常重要的作用。譬如可以利用注释屏蔽一条或者一段语句,以观察程序的变化,发现问题和错误,需要时再去掉注释符即可。

1.1.6 C语言和C++有什么关系?

C++是从C语言发展而来的,它继承了C语言所有的特征,这使得以前的大量C语言代码可以在C++环境中重新编译利用;但是C++也同时从其他面向对象的语言如Ada,Simula67等借鉴了某些面向对象的特征,从而提供了对面向对象编程(Oriented Object Programming,OOP)的支持。

1.1.7 为什么编译系统要提供标准库函数,如何使用标准库函数?

C++程序是由函数组成的,每个函数完成一个独立的功能,一般在写程序时也是将一个独立的功能在一个函数中完成;有些功能是通用的,如数学中的求平方根(sqrt函数),或将一个数输出到屏幕上(cout函数)或者从键盘接收一个数(cin函数)等,C++的编译系统一般都自己提供,这样使用该C++编写程序的程序员就不需要再次写代码来实现这些功能,直接调用系统提供的函数就可以完成。

自己写的代码往往需要经过很多测试才能保证正确性;这些C++系统自身提供的函数一般都包含在随编译器发布的标准库中,因此这些函数称为标准库函数。

要使用标准库中的函数,需要:

(1) 在程序开始处包含相应函数的声明;

(2) 在程序中调用函数;

(3) 将编译生成的目标文件和标准库函数所在的库文件(一般扩展名为.lib)一起连接即可。

系统已经将所有标准库函数的声明分类放到不同的扩展名为.h的文件(称为头文件)中,如大部分数学计算函数的声明都放到math.h,输入/输出函数或对象都放到iostream.h文件中,等等。如果要求一个数的平方根,只要在源程序开始处加#include <math.h>,以后在该文件的源程序中就可以调用sqrt函数。

学习C++,不但要学习语言本身的语法和相关的语句,而且还要学习某些随编译器一起提供的标准库函数的用法,因为C++本身是很小的,很多功能它都不提供,比如输入/输

出等，而是通过调用标准库函数来实现这些功能。尽可能地使用标准库提供的函数而不是自己“另起炉灶”，也可以增加程序的健壮性，提高程序开发的效率。

1.1.8 C++程序的构成和书写形式是什么？

一个C++程序可以由一个程序单位或多个程序单位构成。每一个程序单位作为一个文件。在程序编译时，编译系统分别对各个文件进行编译，因此，一个文件是一个编译单元。在一个程序单位中，可以有以下几个部分：

(1) 预处理命令。一般C++程序中都包括有include命令。

(2) 全局声明部分。(在函数外的声明部分) 这部分中包括对用户自己定义的数据类型的声明和程序中所用到的变量的定义。

(3) 函数。函数是程序中必须有的和最基本的组成部分，它是实现操作的部分。每个程序必须包括一个或多个函数，其中必须有一个(并且只能有一个)主函数(main函数)。

以上这三个部分并不是每个程序文件都必须有的，可以缺少某些部分(包括函数)。也就是说，有的程序文件可以不包括函数，只包括预处理命令和/或声明部分，完全根据需要而定。在C++的编程实践中这一点会体会到的。

一个函数的组成部分包括：

① 函数首部，即函数的第一行。包括函数名、函数类型、函数属性、函数参数(形参)名、参数类型。

例如：

```
int     max  (   int          x  ,      int          y  )
 ↓       ↓       ↓           ↓          ↓           ↓
函数类型 函数名 函数参数类型 函数参数名 函数参数类型 函数参数名
```

一个函数名后面必须有一对圆括号，函数的参数可以默认，如int main( )。

② 函数体，这部分包括在函数首部下面的大括号内。如果在一个函数中有多个大括号，则最外的一对大括号为函数体的范围。函数体一般包括：

- 局部声明部分(在函数内的声明部分)，包括对本函数中所用到的类型、函数的声明和变量的定义。对数据的声明可以放在函数之外(其作用范围是全局的)，也可以放在函数之内(其作用范围是局部的，只在本函数内有效)。
- 执行部分。这部分由若干个执行语句组成，用来进行有关操作，从而实现函数的功能。当然在有些情况下也可以没有声明部分。甚至可以既无声明部分，也无执行部分。如：

```
void dump( ){ }
```

这是一个空函数，什么也不做，但这是合法的。

语句也包括两类：

① 声明语句。如“int a, b”，用来向编译系统通知某些信息(如类型、函数和变量的声明或定义)，但它并不引起实际的操作，属于非执行语句。

② 执行语句。用来实现用户指定的操作，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，显然，没有语句的函数是没有意义的。C++语句必须以英文状态下的分号结束，如“c=a*b;”。

一个C++程序总是从main函数开始执行的，而不论main函数在整个程序的什么位

置。类(Class)是C++新增加的重要数据类型,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封装、信息隐蔽、继承、派生、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(private)和公有(public)属性。私有的数据成员和成员函数只能被本类的成员函数调用。

C++程序的书写格式自由,一行内可以写几个语句,一个语句可以分成若干行写。C++语句没有行号,这一点不同于FORTRAN或COBOL。一个良好的、有使用价值的源程序都应加上必要的注释,这样可以增加程序的可读性。在C++中注释行的标志为"//"或者为"/*……*/","//"只能注释单行,不可以跨行注释;"/*……*/"可以进行跨行注释,在开始注释处用"/*",在注释结束时用"*/"。

1.1.9 如何理解程序设计的目标在正确的前提下,其重要性排列次序依次为:可读、可维护、可移植和高效率?

正确是指程序要能够正确运行起来,并且能够得到正确的结果。

可读是指使用良好的书写风格和易懂的语句编写程序,让团队的其他成员或后继者容易看懂程序的思路和程序代码的含义,知道程序在做什么。

可维护是指当系统的功能需求发生变化时,不需要消耗过多的人力和物力,就可在原来程序代码的基础上增加新的功能,而不需要对原来已经有的代码做过多的修改。

可移植是指不需要对系统做过多的修改就可以将编写的程序从一种环境下放到另外一种环境下运行(可以是在不同的操作系统或不同的硬件环境),而且运行结果一样。

高效率是指软件在运行时,对时间和空间的需求尽量小,比如需要的内存小,时间短等。

软件的正确性是首先要保证的,否则程序也就没有作用。

软件的生命周期包括分析、设计、编码、维护等,程序在编码完成后,才完成整个生命周期的一小阶段,后面极有可能要经常增加新的功能或者矫正程序在编码过程中没有发现,却在运行中出现的问题;而现在的项目开发大多是团队合作(因为现在的软件系统越来越大,功能越来越复杂,不可能是一个人单独完成),因此修改和增加功能的程序员不一定就是原来编码的程序员,不理解程序的代码就很难修改程序,因此最初编码的程序员要使自己的代码容易阅读,在程序代码的适当的地方提供注释,当然要容易读懂程序还必须要提供相关的开发时的文档等。

要程序容易维护,必须要在软件设计阶段就注意,整个软件的结构要设计合理,才能够做到可维护,或者容易维护。

为了提高程序的可读性,往往会在程序中产生重复的代码,这有时对程序的效率会产生一些影响,但随着硬件性价比的提高,可以不需花费太多费用,就可大幅度提高机器的性能,从而抵消代码的影响。除非是特别底层的系统软件,一般是对效率不需花费太多的精力,不需要费尽心机去寻找一个优秀的算法,这样也可以提高开发的效率。

1.1.10 什么是数值溢出?整数溢出与浮点数溢出有何不同?

程序中当一个数值变量的值对于它的类型来说太大时,会发生数值溢出。例如,大多数计算中,short型的变量不能超过32767,因此当一个这种类型的变量值为32767时,如果再被加(或通过其他任何算法操作被加),则会发生溢出。当发生整数溢出时,这个变量会被

“反卷”成负数，产生错误的结果。当发生浮点数溢出时，这个变量的值会被设为代表无穷大的常数 inf。

1.2 典型题解

题型 1 基础知识

【例 1-1】 关于 C++和 C 语言的关系的下列描述中，错误的是_____。

A. C 语言是 C++的一个子集　　B. C 语言和 C++都是面向对象的语言

C. C++与 C 语言兼容　　D. C++对 C 语言做了些改进

分析：本题考查 C++和 C 语言的关系。C 是 C++的一个子集，C++在 C 的基础上添加了面向对象的语言特征，从而支持面向对象编程，但是 C 不是面向对象的语言。

解答：B

评注：C++是在 C 语言的基础上发展起来的，在 C 语言的基础上添加了很多支持面向对象编程的语言特征，从而使 C++成为一种面向对象的程序设计语言，但是正是因为 C++是从 C 发展过来的，而且为了使以前成千上万程序员已经写的大量优秀的 C 代码不至于浪费掉，C++兼容了 C 的所有特征，使得以前 C 代码编写的程序可以在 C++环境中编译和运行。

【例 1-2】 下面关于面向对象的描述正确的是_____。

A. 面向对象是一种编程思想

B. 面向对象就是使用对象来模拟现实中的事物

C. 面向对象将描述事物的数据和操作这些事物的操作封装在一起，构成对象

D. 使用面向对象编程比面向结构编程更能提高程序员的工作效率

解答：ABCD

评注：本题考查面向对象的概念。面向对象就是创建和现实问题中相对应或者相近的对象，在计算机中通过这些对象之间的动态交互来模拟现实世界中的问题。面向对象本身是一种编程思想，各种语言实现面向对象思想的具体技术和方式可能不一样。

【例 1-3】 下面描述正确的是_____。

A. C++是一种面向结构化程序设计的语言

B. C++是一种面向对象的程序设计的语言

C. C++是一种通用的程序设计语言

D. C 语言是一种面向对象的程序设计语言

分析：本例主要考查一种语言对程序设计方法的支持的理解。C++提供了一系列的语言特征，如类，继承，多态性来支持面向对象的程序设计，因此它是面向对象程序设计的语言。

C++同时也提供了函数、名字空间、模板等来支持面向过程、面向结构、泛型程序设计，因此称 C++是一种通用的程序设计语言。而 C 语言没有提供语言特征来方便的进行面向对象的编程，因此 C 不是一种面向对象的程序设计语言。

解答：ABC

评注：说一个语言支持某种程序设计方法，它必须提供了一些语言特征，使得它能够方便（比较容易、安全和有效地）用于这种程序设计风格，以及对无意中偏离了这种范型的情况做编译或者运行时的检查；如果要写那样的程序必须付出很大的努力或者利用各种技巧，就说一个语言不支持某种程序设计技术。