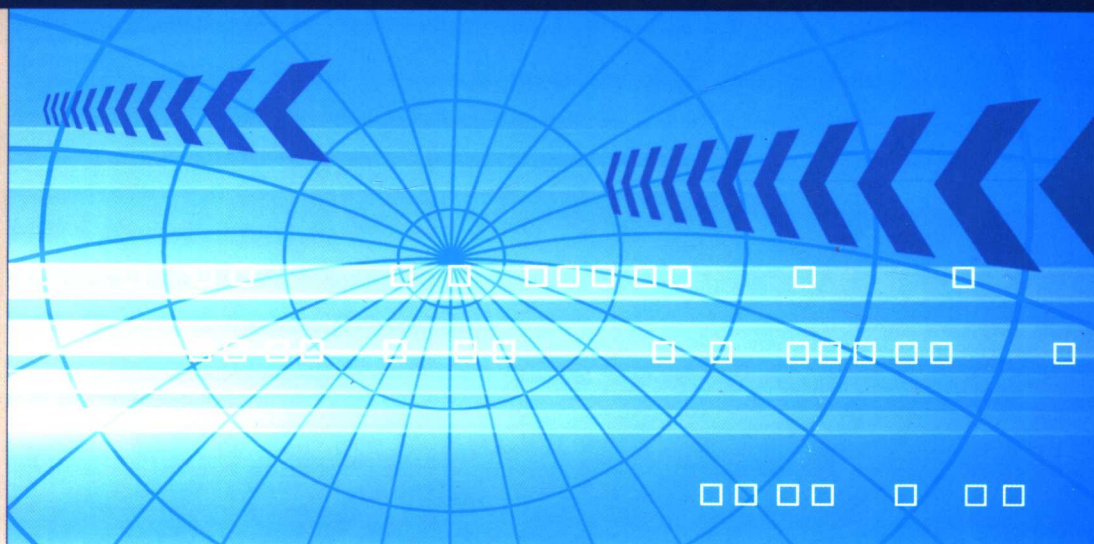




高职高专计算机技能型紧缺人才培养规划教材

计算机软件技术专业

主编 张晓蕾 蒋方纯



# 数据结构与算法

张晓蕾 编



 人民邮电出版社  
POSTS & TELECOM PRESS

高职高专计算机技能型紧缺人才培养规划教材  
计算机软件技术专业

# 数据结构与算法

---

张晓蕾 编 ←

人民邮电出版社

## 图书在版编目 (CIP) 数据

数据结构与算法/张晓蕾编. —北京: 人民邮电出版社, 2005.7

ISBN 7-115-13319-0

I. 数... II. 张... III. ①数据结构②算法分析 IV. TP311.12

中国版本图书馆 CIP 数据核字 (2005) 第 054097 号

### 内 容 提 要

数据结构是计算机软件相关专业的核心课程之一, 主要讲授数据结构和算法的原理、方法和技术。从研究对象来说, 包括向量、链表、矩阵、堆栈、队列、串、树和图等; 从常用算法来说, 包括查找和排序等。

本书的每一章都从数据结构的基本思想出发, 一直讲授到标准模板库 (STL) 的有关内容。通过本书的学习, 读者可以熟悉 C++ 开发技术, 特别是面向对象的程序设计; 掌握算法设计和数据结构设计的基本方法; 掌握如何评价、选择和使用各种库函数所实现的算法; 理解并学会使用 STL。

本书可作为高职高专院校计算机软件专业教材, 也可作为计算机软件专业本科学生的参考书, 对于从事计算机软件开发的技术人员也有一定参考价值。

高职高专计算机技能型紧缺人才培养规划教材

计算机软件技术专业

数据结构与算法

- 
- ◆ 编 者 张晓蕾  
责任编辑 潘春燕
  - ◆ 人民邮电出版社出版发行 北京市崇文区夕照寺街 14 号  
邮编 100061 电子函件 315@ptpress.com.cn  
网址 <http://www.ptpress.com.cn>  
北京隆昌伟业印刷有限公司印刷  
新华书店总店北京发行所经销
  - ◆ 开本: 787×1092 1/16  
印张: 19.5  
字数: 463 千字 2005 年 7 月第 1 版  
印数: 1—3 000 册 2005 年 7 月北京第 1 次印刷

---

ISBN 7-115-13319-0/TP · 4617

定价: 26.00 元

读者服务热线: (010) 67170985 印装质量热线: (010) 67129223

# 高职高专计算机技能型紧缺人才培养

## 规划教材编委会

主 任 武马群

副主任 王泰峰 徐民鹰 王晓丹

编 委 (以姓氏笔画为序)

马 伟	安志远	向 伟	刘 兵	吴卫祖	吴宏雷
余明辉	张晓蕾	张基宏	贺 平	柳 青	赵英杰
施晓秋	姜 锐	耿 壮	郭 勇	曹 炜	蒋方纯
潘春燕					

## 丛书出版前言

目前,人才问题是制约我国软件产业发展的关键。为加大软件人才培养力度和提高软件人才培养质量,教育部继在2003年确定北京信息职业技术学院等35所高职院校试办示范性软件职业技术学院后,又同时根据《教育部等六部门关于实施职业院校制造业和现代服务业技能型紧缺人才培养培训工程的通知》(教职成[2003]5号)的要求,组织制定了《两年制高等教育计算机应用与软件技术专业领域技能型紧缺人才培养指导方案》。示范性软件职业技术学院与计算机应用与软件技术专业领域技能型紧缺人才培养工作,均要求在较短的时间内培养出符合企业需要、具有核心技能的软件技术人才,因此,对目前高等职业教育的办学模式和人才培养方案等做较大的改进和全新的探索已经成为学校的当务之急。

据此,我们认为做一套符合上述一系列要求的切合学校实际的教学方案尤为重要。遵照教育部提出的以就业为导向,高等职业教育从专业本位向职业岗位和就业为本转变的指导思想,根据目前高等职业教育院校日益重视学生将来的就业岗位,注重培养毕业生的职业能力的现状,我们联合北京信息职业技术学院等几十所高职院校和普拉内特计算机技术(北京)有限公司、福建星网锐捷网络有限公司、北京索浪计算机有限公司等软件企业共同组建了计算机应用与软件技术专业领域技能型紧缺人才培养教学方案研究小组(以下简称研究小组)。研究小组对承担计算机应用与软件技术专业领域技能型紧缺人才培养培训工作的79所院校的专业设置情况做了细致的调研,并调查了几十所高职院校计算机相关专业的学生就业情况以及目前软件企业的人才市场需求状况,确定首批开发目前在高职院校开设比较普遍的计算机软件技术、计算机网络技术、计算机多媒体技术和计算机应用技术等4个专业方向的教学方案。

同时,为贯彻教育部提出的要与软件企业合作开展计算机应用与软件技术专业领域技能型紧缺人才培养培训工作的精神,使高等职业教育培养出的软件技术人才符合企业的需求,研究小组与许多软件企业的专家们进行了反复研讨,了解到目前高职院校的毕业生的实际动手能力和综合应用知识方面较弱,他们和企业需求的软件人才有着较大的差距,到企业后不能很快独当一面,企业需要投入一定的成本和时间进行项目培训。针对这种情况,研究小组在教学方案中增加了“综合项目实训”模块,以求强化学生的实际动手能力和综合应用前期所学知识的能力,探索将企业的岗前培训内容前移到学校的教学中的实验之路,以此增强毕业生的就业竞争力。

在上述工作的基础上,研究小组于2004年多次组织召开了包括企业专家、教育专家、学校任课教师在内的各种研讨会和方案论证会,对各个专业按照“岗位群→核心技能→知识点→课程设置→各课程应掌握的技能→各教材的内容”一步步进行了认真的分析和研讨:

- 列出各专业的岗位群及核心技能。针对教育部提出的以就业为导向,根据目前高职高专院校日益关心学生将来的就业岗位的现状,在前期大量调研的基础上,首先提炼各个专业的岗位群。如对某专业的岗位群进行研究时,首先罗列此专业的各个岗位,以便能正确了解

每个岗位的职业能力，再根据职业能力进行有意义的合并，形成各个专业的岗位群，再对每个岗位群总结和归纳出其核心技能。

- 根据岗位群及核心技能做出教学方案。在岗位群及核心技能明确的前提下，列出此岗位应该掌握的知识点，再依据这些知识点推出应该学习的课程、学时数、课程之间的联系、开课顺序并进行必要的整合，最终形成一套科学完整的教学方案。

为配合学校对技能型紧缺人才的培养工作，在研究小组开发上述4个专业的教学方案的基础上，我们组织编写了这套包含计算机软件技术、计算机网络技术、计算机多媒体技术及计算机应用技术等4个专业的教材。本套教材具有以下特点：

- 注重专业整体策划的内涵。对各专业系列教材按照“岗位群→核心技能→知识点→课程设置→各课程应掌握的技能→各教材的内容”的思路组织开发教材。

- 按照“理论够用为度”的原则，对各个专业的基础课进行了按需重新整合。

- 各专业教材突出了实训的比例，注重案例教学。每本教材都配备了实验、实训的内容，部分专业的教材配备了综合项目实训，使学生通过模拟具体的软件开发项目了解软件企业的运行环境，体验软件的规范化、标准化、专业化和规模化的开发流程。

为了方便教学，我们免费为选用本套教材的老师提供部分专业的整体教学方案及教学相关资料。

- 所有教材的电子教案。
- 部分教材的习题答案。
- 部分教材中实例制作过程中用到的素材。
- 部分教材中实例的制作效果以及一些源程序代码。

本套教材以各个专业的岗位群为出发点，注重专业整体策划，试图通过对系列教材的整体构架，探索一条培养技能型紧缺人才的有效途径。

经过近两年的艰苦探索和工作，本套教材终于正式出版了，我们衷心希望，各位关心高等职业教育的读者能够对本套教材的不当之处给予批评指正，提出修改意见，也热切盼望从事高等职业教育的教师以及软件企业的技术专家和我们联系，共同探讨计算机应用与软件技术专业的教学方案和教材编写等相关问题。来信请发至 [panchunyan@ptpress.com.cn](mailto:panchunyan@ptpress.com.cn)。

## 编者的话

数据结构与算法是软件开发的核⼼技术之一，它是研究各种数据结构的特点、存储结构及相关算法的学科。随着计算机科学的快速发展，数据结构不断发展，并且越来越与计算机编程紧密结合，贯穿程序设计的始终。计算机专家曾将程序定义为“程序=数据结构+算法”，这也说明了数据结构和算法的重要性。

大学中开设的数据结构课程由于理论性较强，往往令学生望⽣畏。本书的编写思路是尽量降低学生学习该课程的难度，为此引入了我们多年进行数据结构课程教学改革与试验的成果，围绕软件人才的培养目标，根据企业需求与教学规律进行了课程的整合，以就业岗位的要求设计课程内容，强调软件系统开发实用技术的教学，注重对学生工程化能力的培养，提高学生的软件开发能力。

本书使用 C++ 标准中新增加的标准模板库 (Standard Template Library, STL) 来直观地讲述算法分析、通用容器类和常用算法，介绍的算法可以直接应用到软件开发实践中，极大地提高软件系统的规范性和开发效率。通过使用模板，C++ 推迟对类型的选择，允许程序员针对不同的类型提取出一般化的代码，相对于传统的、由函数及类组成的库来说，模板库可以提供更好的代码重用机会，所以它被国际标准化组织添加到 C++ 标准中。本书一方面采用 C++ 语言描述数据结构和算法，简洁地讲解数据结构的 C++ 表示、算法的基本思想；另一方面讲述 STL 的容器与算法以及常用的通用算法，力求在基本知识与 STL 描述之间取得平衡，做到描述直观和易于理解。另外，本书还具有以下特色。

(1) 第 1 章给出了 C++ 基本特征和面向对象的程序设计的基础知识。

(2) 每章的算法从问题分析出发，着重讲解解决问题的思路、程序设计的要点，并尽量给出直观的图示。

(3) 每章都利用介绍的各种算法编写相应的 C++ 类，并放在指定的头文件中。

(4) 每章利用输入类明确地给出调用算法的输入数据。

(5) 根据讲述内容的展开，系统介绍 STL 的基本概念、容器、迭代器和通用算法。

(6) 每章都包含丰富的例题，供学生程序设计参考。

(7) 每章都配备适当的习题，供学生复习基本知识和按指定要求编写 C++ 程序。

本书为高职高专计算机软件专业教材，讲授学时为 70 学时左右，对其他专业（如计算机应用专业），可以根据专业特点和学生的实际情况做适当取舍。容器类算法及标准模板库通用算法的实践性很强，上机实验是必不可少的教学环节。因此，必须安排适量的上机实践，建议其学时数不少于讲授学时的 1/4。

参与本书编写的主要有张晓蕾、张斐慕、陈炜、朱立、万忠、苏飞、杨旭、魏洪涛、马世珍等，在大纲编写、教材审定以及在试用的过程中，有关教授、专家和软件公司提出了许多宝贵的意见，在此一并感谢。

要编写好一本教材是不容易的，要编写好一本从基本知识到 STL 的数据结构教材更不容易。受作者水平所限，本书难免有不少缺点和不足，希望得到专家、计算机同行和广大读者的批评指正。

编者  
2005.4



# 目 录

<b>第 1 章</b>	<b>面向对象程序设计基础</b> .....	1
1.1	C++基本特征.....	1
1.1.1	重载.....	1
1.1.2	缺省参数函数与内置函数.....	3
1.1.3	引用及其使用.....	3
1.1.4	动态内存分配.....	7
1.2	类和对象.....	8
1.2.1	声明类和对象.....	9
1.2.2	在类外定义成员函数.....	10
1.2.3	类数组、类指针与函数的类参数.....	11
1.2.4	this 指针.....	14
1.3	构造函数与析构函数.....	15
1.3.1	构造函数.....	15
1.3.2	析构函数.....	16
1.4	继承与派生.....	17
1.4.1	建立派生类.....	18
1.4.2	公用派生类与私有派生类.....	19
1.4.3	保护成员.....	20
1.4.4	友元函数与友元类.....	20
1.5	C++模板.....	23
1.5.1	函数模板.....	23
1.5.2	类模板.....	25
	习题.....	26
<b>第 2 章</b>	<b>数据结构导论</b> .....	31
2.1	数据结构的基本概念.....	31
2.1.1	几个实例.....	31
2.1.2	数据结构的术语.....	33
2.1.3	抽象数据类型及其实现.....	34
2.2	算法描述.....	35
2.2.1	算法的特性.....	35
2.2.2	算法描述与通用性.....	35
2.2.3	类和算法的测试.....	38

2.3 C++标准模板库简介	38
2.4 算法分析初步	41
习题	44
<b>第 3 章 向量</b>	<b>47</b>
3.1 向量的基本知识	47
3.1.1 线性表的定义	47
3.1.2 向量的存储结构	47
3.2 向量运算	48
3.2.1 向量运算简介	48
3.2.2 插入算法与删除算法	49
3.3 简易向量类	52
3.3.1 简易向量类及其实现	52
3.3.2 简易向量类的测试	54
3.4 标准模板向量类	55
3.4.1 模板向量类的构造器及下标运算符	55
3.4.2 模板向量类的迭代器	56
3.4.3 模板向量类的成员函数	58
3.4.4 insert 类算法和 erase 类算法分析	58
3.4.5 模板向量类的一般表示	61
3.5 模板向量容器的测试类	62
3.5.1 模板向量测试类的数据输入	62
3.5.2 模板向量测试类的源代码	63
3.5.3 模板向量测试类的使用	66
3.6 矩阵类	67
3.6.1 矩阵容器的描述	68
3.6.2 模板矩阵类的使用	69
习题	71
<b>第 4 章 链表</b>	<b>73</b>
4.1 链表存储结构的基本知识	73
4.1.1 单链表与指针	73
4.1.2 单链表的基本运算	75
4.2 简易的单链表类	79
4.2.1 单链表类源代码	79
4.2.2 单链表类的测试	80
4.3 循环链表和双向链表	82
4.3.1 循环链表	82
4.3.2 双向链表	82

4.4 标准模板双向链表类 .....	83
4.4.1 模板链表类的一般表示 .....	84
4.4.2 构造器 .....	84
4.4.3 模板链表类的迭代器 .....	85
4.4.4 模板链表类的成员函数 .....	87
4.5 模板链表容器的测试类 .....	89
4.5.1 模板链表测试类的数据输入 .....	89
4.5.2 模板链表测试类的源代码 .....	89
4.5.3 模板链表测试类的使用 .....	93
习题 .....	95
<b>第 5 章 栈和队列</b> .....	<b>97</b>
5.1 栈 .....	97
5.1.1 栈的定义及运算 .....	97
5.1.2 栈的向量存储结构 .....	98
5.1.3 简易向量栈类 .....	98
5.1.4 栈的链表存储结构 .....	102
5.1.5 简易链表栈类 .....	103
5.2 模板栈容器 .....	108
5.2.1 模板栈容器的实现 .....	108
5.2.2 栈的接口 .....	109
5.2.3 使用栈容器 .....	110
5.3 队列的基本知识 .....	115
5.3.1 队列的定义及运算 .....	115
5.3.2 队列的向量存储结构 .....	116
5.3.3 简易的循环队列类 .....	117
5.3.4 队列的链表存储结构 .....	120
5.4 模板队列容器 .....	124
5.4.1 deque 容器 .....	124
5.4.2 queue 容器 .....	125
习题 .....	128
<b>第 6 章 字符串</b> .....	<b>133</b>
6.1 字符与字符串的概念 .....	133
6.2 C 风格字符串的存储结构与运算 .....	134
6.2.1 C 风格字符串的顺序存储结构 .....	134
6.2.2 C 风格字符串的链表存储结构 .....	135
6.2.3 C 风格字符串的运算 .....	136
6.3 字符串类 .....	138

6.3.1	字符串类对象的声明	139
6.3.2	字符串类的构造器	140
6.3.3	字符串类的运算	141
6.4	模板容器与字符串类	144
	习题	148

## 第 7 章 查找 151

7.1	查找方法概述	151
7.2	顺序查找	152
7.2.1	简单顺序查找方法	152
7.2.2	一般线性表的顺序查找	153
7.3	有序表的查找	156
7.3.1	有序表的建立	156
7.3.2	有序表的折半查找法	157
7.3.3	折半查找法的应用	161
7.4	标准模板库的查找算法	164
7.4.1	STL 迭代器	164
7.4.2	STL 查找运算	165
7.5	哈希表及其查找	169
7.5.1	哈希表与哈希函数	169
7.5.2	设计哈希函数	171
7.5.3	闭散列方法	173
7.5.4	哈希类的向量版本	176
7.5.5	开散列方法与哈希类的链表版本	179
	习题	183

## 第 8 章 排序 187

8.1	排序基本概念	187
8.2	3 种基本的排序方法	188
8.2.1	直接插入排序	188
8.2.2	冒泡排序	189
8.2.3	选择排序	191
8.2.4	基本排序方法的向量版本	192
8.3	高级排序方法	194
8.3.1	希尔排序	194
8.3.2	快速排序	195
8.3.3	归并排序	198
8.3.4	基数排序	200
8.4	标准模板库的通用排序方法	202

习题	209
<b>第 9 章 树</b>	<b>212</b>
9.1 二叉树	212
9.1.1 树的常用术语	212
9.1.2 二叉树的定义	213
9.1.3 二叉树的重要性质	213
9.1.4 二叉树的存储结构	214
9.2 遍历二叉树	215
9.2.1 先根遍历	216
9.2.2 中根遍历	217
9.2.3 后根遍历	218
9.2.4 按层遍历	219
9.2.5 二叉树遍历算法的应用	220
9.3 二叉链表模板类	221
9.3.1 二叉链表类的输入类	222
9.3.2 二叉链表类的模板结点类	222
9.3.3 二叉链表基类	223
9.3.4 二叉链表类派生的应用类	228
9.4 二叉搜索树模板类	231
9.4.1 二叉搜索树的基本知识	231
9.4.2 定位函数与查找算法的实现	235
9.4.3 二叉搜索树类及其测试	237
9.4.4 二叉搜索树插入算法与删除算法的实现	240
9.4.5 输出二叉树结点表算法的实现	245
习题	248
<b>第 10 章 图</b>	<b>250</b>
10.1 图的基本概念和术语	250
10.1.1 图的概念	250
10.1.2 路径和回路	251
10.1.3 连通图	251
10.1.4 顶点的度	252
10.2 图的存储结构	252
10.2.1 邻接矩阵	252
10.2.2 基于邻接矩阵的模板图类	253
10.2.3 邻接链表	258
10.2.4 基于邻接链表的模板图类	259
10.3 图的遍历及其应用	264

10.3.1	图的深度优先搜索遍历 .....	265
10.3.2	图的广度优先搜索遍历 .....	268
10.3.3	图遍历的应用 .....	269
10.3.4	图的遍历类 .....	272
10.4	图的生成树 .....	276
10.4.1	生成树的概念 .....	276
10.4.2	最小生成树 .....	277
10.4.3	Prim 算法 .....	280
10.4.4	Kruskal 算法 .....	284
10.5	最短路径 .....	288
10.5.1	单源顶点最短路径问题求解 .....	289
10.5.2	关于源点最短路径的讨论 .....	292
	习题 .....	294

# 第 1 章

## 面向对象程序设计基础

面向对象的程序设计 (Object-Oriented Programming, OOP) 是软件工程研究的重要领域, 是为解决复杂问题而提出来的, 代表了当今国际上软件程序设计的潮流。OOP 打破了传统的代码、数据分离的做法, 是全新的程序设计方法。在 Windows 操作系统取得巨大成功之后, 几乎所有的 Windows 软件开发工具都是基于面向对象程序设计的。

编写面向对象的程序, 能够使开发者重复使用自己的工作成果, 并且使程序的维护更为容易、程序的扩展更为方便, 大大减少修改所带来的错误。

本章主要介绍 C++ 的基本特征, 并在此基础上介绍面向对象的程序设计, 在 C++ 语言中, OOP 主要包括 3 个基本技术: 类、继承和多态性。根据本书的需要, 我们仅介绍前两个基本技术。阅读本章内容前, 读者需掌握 C 语言, 并对 C++ 语言有初步了解。通过本章的学习, 读者应该掌握以下内容:

- 函数重载与运算符重载
- 引用及其使用, 特别是引用作为函数参数
- 动态内存分配运算符 new 和清除内存运算符 delete
- 类的定义和对象的声明
- 使用类的构造函数与析构函数
- 类的继承机制
- 使用友元函数与友元类
- 书写和使用函数模板和类模板

### 1.1 C++ 基本特征

本节介绍 C++ 语言对 C 语言的扩充, 我们称这些扩充为 C++ 的基本特征。

#### 1.1.1 重载

##### 1. 函数重载

用 C 语言编程时, 有时会出现这种情况: 几个不同名的函数, 实现的是同一类操作。例如, 要从两个数中找出较大的数, 而这两个数可以是整型、实型或字符型, 由于 C 语言规定在同一作用域中不能有同名函数, 因此程序员必须设计 3 个不同的函数。

在 C++ 中, 只要函数的参数个数或者参数类型不同, 就允许在同一作用域中出现同名函数, 这就是函数的重载。不难看出, C++ 中重载的含义是: 参数不同的同名函数认为是不同的函数。

**例 1-1** 编写同名 max() 函数。

对上面的问题，可以在同一作用域中编写同名的 max() 函数，以求两个数中的较大者（分别考虑整型、实型和字符型）。

这 3 种 max() 函数的原型分别是：

```
int max(int a, int b);           // 求两个整数中的较大者
float max(float a, float b);    // 求两个实数中的较大者
char max(char a, char b);       // 求两个字符中的较大者
```

它们的函数体都是下面的条件语句：

```
if (a>b) return a;
else     return b;
```

**2. 运算符重载**

在 C++ 中，运算符也可以重载。也就是说，允许一个运算符用于不同场合、有不同的含义。例如，插入运算符 “<<” 和提取运算符 “>>”，本来是位运算的左移运算符和右移运算符，在 C++ 中又作为输出 / 输入运算符。又如，结构体之间是没有比较运算的，为表示简练，在有些算法中常用结构体关键字作为它们比较大小的依据。

**例 1-2** 重载结构体的 “<” 运算符。

可以用如下方法比较两个结构体的大小：比较结构体关键字的大小。重载 “<” 运算符，并利用它输出两个结构体中的较大者，程序代码如下。

```
struct TNode
{
    int key;           // 关键字
    char ch;          // 结构体其他数据成员
};

bool operator>(const TNode x, const TNode y)
{
    return (x.key > y.key);
}

TNode max(TNode a, TNode b)           // 求两个结构体中的较大者
{
    if (a>b) return a;
    else     return b;
}

void main()
{
    struct TNode a = {31,'c'}, b={104,'h'}, node;
    node = max(a,b);
}
```



```
cout << "关键字较大结构体是: 《" << node.key << ", " << node.ch  
    << "》 " << endl;  
}
```

程序运行结果如下。

关键字较大结构体是: 《104, h》

### 1.1.2 缺省参数函数与内置函数

#### 1. 缺省参数函数

一般情况下, 实参个数应与形参个数相同, 但 C++ 允许实参个数与形参个数不同。使用方法是在形参表中对一个或几个形参指定缺省值(或默认值)。例如, 函数原型可用如下形式:

```
void func(int a, int b, int c=100)
```

在调用这个函数时, 如果写成 `func(2, 4, 6)`, 则形参 `a`、`b`、`c` 的值分别为 2、4、6。如果写成 `func(2, 4)`, 则形参 `a`、`b`、`c` 的值分别为 2、4、100。

**注意:** 赋予缺省值的参数必须放在形参表的最右端。例如:

```
void func(float a, int b, int c=0, char d='a')
```

利用这一特性, 可以使函数的使用更加灵活。例如, 如果已编写了求 3 个整数中最大者的函数:

```
int max(int a, int b, int c)
```

则若要用它求两个整数中的较大者, 只需将函数修改为:

```
int max(int a, int b, int c = -2147483648)
```

#### 2. 内置函数

如果有的函数需要频繁调用, 为提高效率, C++ 提供一种方法可将这种函数的代码在编译时嵌入到主调函数中。这种嵌入到主调函数中的函数称为内置函数(或内联函数)。

指定内置函数的方法很简单, 只需在函数原型的左端增加一个 `inline` 关键字。例如, 声明求 3 个整数中的最大者的内置函数原型如下:

```
inline int max(int a, int b, int c)
```

### 1.1.3 引用及其使用

引用(Reference)是 C++ 的一种新的变量类型, 是对 C 语言的一个重要扩充。要注意引用与指针的取地址符的区别。

引用是给对象起一个别名。引用的说明符如下:

```
类型名 &
```

其含义是指定类型的引用, 即该引用与指定类型的变量地址相联系。例如:

```
int a;
```

```
int &b = a;
```

上面的语句创建了整型类型的引用 `b`, `b` 是 `a` 的引用, 即 `a` 的“别名”。经过这样的声明后, 使用 `a` 或 `b` 的作用完全相同, 都代表同一变量。

**注意:** 在上面的声明中, “&”是引用声明符, 而不是地址符。不要理解为“把 `a` 的值赋给 `b` 的地址”。