<<C语言的科学和艺术>>

图书基本信息

书名：<<C语言的科学和艺术>>

13位ISBN编号：9787111347750

10位ISBN编号：7111347757

出版时间：2011-6

出版时间：机械工业出版社

作者：罗伯茨

页数：538

译者：翁惠玉,张冬茉,杨鑫,蒋文新

<<C语言的科学和艺术>>

内容概要

《C语言的科学和艺术》是计算机科学的经典教材，介绍了计算机科学的基础知识和程序设计的专门知识。
《C语言的科学和艺术》以介绍ANSI
C为主线，不仅涵盖C语言的基本知识，而且介绍了软件工程技术以及如何应用良好的程序设计风格进行开发等内容。
《C语言的科学和艺术》采用了库函数的方法，强调抽象的原则，详细阐述了库和模块化开发。
此外，《C语言的科学和艺术》还利用大量实例讲述解决问题的全过程，对开发过程中常见的错误也给出了解决和避免的方法。
《C语言的科学和艺术》既可作为高等院校计算机科学入门课程及C语言入门课程的教材，也是C语言开发人员的极佳参考书。

作者简介

Eric S.
Roberts第一次教授计算机科学入门课程是在20多年前，当时他还是一名哈佛学子。
1980年获得博士学位以来，Roberts先后在哈佛、韦尔斯利、斯坦福大学教授计算机科学课程。
在斯坦福大学，Roberts担任计算机科学系副主任，并负责本科生的计算机科学课程。
虽然Roberts也教授计算机科学的高级课程并作一些研究工作，但他最大的乐趣还是向初学者展示计算机的强大功能。

<<C语言的科学和艺术>>

书籍目录

译者序
前言
第1章 概述
1.1 计算简史
1.2 什么是计算机科学
1.3 计算机硬件简介
1.3.1 cpu
1.3.2 内存
1.3.3 辅助存储器
1.3.4 i/o设备
1.4 算法
1.5 程序设计语言和编译
1.6 编程错误和调试
1.7 软件维护
1.8 软件工程的重要性
1.9 关于c程序设计语言的一些思考
小结
复习题
第一部分 c语言程序设计基础
第2章 通过例子学习
2.1 “ hello world ”程序
2.1.1 注释
2.1.2 库包含
2.1.3 主程序
2.2 两个数的加法程序
2.2.1 输入阶段
2.2.2 计算阶段
2.2.3 输出阶段
2.3 有关程序设计过程的观点
2.4 数据类型
2.4.1 浮点型数据
2.4.2 字符串类型的数据
2.5 表达式
2.5.1 常量
2.5.2 变量
2.5.3 赋值语句
2.5.4 运算符和操作数
2.5.5 整型数和浮点型数的结合
2.5.6 整数除法和求余运算符
2.5.7 优先级
2.5.8 优先级法则的应用
2.5.9 类型转换
小结
复习题
程序设计练习

第3章 问题求解
3.1 程序设计习语和范例
3.1.1 复合赋值习语
3.1.2 自增和自减运算符
3.2 解决规模稍大的问题
3.3 控制语句
3.3.1 重复n次习语
3.3.2 迭代和循环
3.3.3 下标变量
3.3.4 初始化的重要性
3.3.5 读入-直到-标志习语
3.3.6 创造一个更实用的应用程序
3.3.7 条件执行和if语句
3.4 一个调试练习
3.5 格式化输出
3.5.1 printf的格式码
3.5.2 控制空格、对齐方式和精度
3.6 构思一个程序
3.6.1 程序设计风格
3.6.2 设计时考虑将来的修改
3.6.3 #define机制
小结
复习题
程序设计练习
第4章 语句形式
4.1 简单语句
4.1.1 赋值的嵌套
4.1.2 多重赋值
4.1.3 程序块
4.2 控制语句
4.3 布尔型数据
4.3.1 关系运算符
4.3.2 逻辑运算符
4.3.3 简化求值
4.3.4 标志
4.3.5 避免布尔表达式中的冗余
4.3.6 布尔计算示例
4.4 if语句
4.4.1 单行if语句
4.4.2 多行if语句
4.4.3 if/else语句
4.4.4 级联if语句
4.4.5 ?: 运算符（可选的）
4.5 switch语句
4.6 while语句
4.6.1 while循环的应用
4.6.2 无限循环

4.6.3 解决半途退出问题
4.7 for语句
4.7.1 嵌套的for循环
4.7.2 for和while的关系
4.7.3 for语句中浮点型数据的使用问题
小结
复习题
程序设计练习
第5章 函数
5.1 使用库函数
5.2 函数声明
5.3 自己编写函数
5.3.1 return语句
5.3.2 将函数与主程序放在一起
5.3.3 包含内部控制结构的函数
5.3.4 返回非数字值的函数
5.3.5 谓词函数
5.3.6 测试字符串是否相等的谓词函数
5.4 函数调用过程机制
5.4.1 参数传递
5.4.2 在其他函数中调用函数
5.5 过程
5.6 逐步精化
5.6.1 从顶开始
5.6.2 实现printcalendar
5.6.3 实现printcalendarmonth
5.6.4 完成最后的片段
小结
复习题
程序设计练习
第6章 算法
6.1 测试素数
6.1.1 一个isprime的简单版本
6.1.2 验证一个策略是否表示一个算法
6.1.3 说明isprime算法的正确性
6.1.4 改进算法的效率
6.1.5 在各个可选方案中选择
6.2 计算最大公约数
6.2.1 brute-force算法
6.2.2 欧几里得算法
6.2.3 欧几里得算法的正确性说明（可选）
6.2.4 比较gcd算法的效率
6.3 数值算法
6.3.1 连续逼近
6.3.2 报告错误
6.4 级数展开
6.4.1 zeno悖论

6.4.2 用级数展开法设计平方根函数
6.4.3 估计平方根的泰勒级数展开 （可选）
6.4.4 泰勒级数近似的实现
6.4.5 停留在收敛半径之内
6.5 指定数值类型的大小
6.5.1 整数类型
6.5.2 无符号类型
6.5.3 浮点类型
小结
复习题
程序设计练习
第二部分 库和模块化开发
第7章 库和接口：一个简单的图形库
7.1 接口的概念
7.2 图形库介绍
7.2.1 graphics.h的基本模型
7.2.2 graphics.h接口的函数
7.2.3 软件包初始化
7.2.4 画直线
7.2.5 画圆和弧
7.2.6 获取有关图形窗口的信息
7.3 建立自己的工具
7.3.1 定义drawbox
7.3.2 定义drawcenteredcircle
7.3.3 绝对坐标和相对坐标间的切换
7.3.4 定义过程的好处
7.4 解决一个较大的问题
7.4.1 使用逐步精化
7.4.2 实现drawhouse过程
7.4.3 寻找共同的模式
7.4.4 结束分解
小结
复习题
程序设计练习
第8章 设计接口：一个随机数库
8.1 接口设计
8.1.1 同一主题的重要性
8.1.2 简单性和信息隐藏的原则
8.1.3 满足客户的需要
8.1.4 通用工具的优势
8.1.5 稳定性的价值
8.2 用计算机生成随机数
8.2.1 确定行为与非确定行为
8.2.2 随机数和伪随机数
8.2.3 ansi c中生成伪随机数
8.2.4 改变随机数的范围
8.2.5 将此问题通用化

8.3 在库中保存工具
8.3.1 接口的内容
8.3.2 写random.h接口
8.3.3 random.c的实现
8.3.4 构造客户程序
8.3.5 初始化随机数发生器
8.4 评价random.h接口的设计
8.4.1 产生随机实数
8.4.2 模拟一个概率事件
8.4.3 在接口中包含头文件
8.4.4 完成随机数软件包的实现
8.5 使用随机数软件包
小结
复习题
程序设计练习
第9章 字符串和字符
9.1 枚举的原理
9.1.1 在机器内部表示枚举类型
9.1.2 将枚举类型表示为整数
9.1.3 定义新的枚举类型
9.1.4 枚举类型的操作
9.1.5 标量类型
9.2 字符
9.2.1 数据类型char
9.2.2 ascii代码
9.2.3 字符常量
9.2.4 ascii代码方案的重要特性
9.2.5 特殊字符
9.2.6 字符运算
9.2.7 ctype.h接口
9.2.8 涉及字符的控制语句
9.2.9 字符的输入输出
9.3 字符串作为抽象数据类型
9.3.1 分层抽象
9.3.2 抽象类型的概念
9.4 strlib.h接口
9.4.1 确定字符串的长度
9.4.2 从一个字符串中选择字符
9.4.3 连接
9.4.4 将字符转换为字符串
9.4.5 抽取字符串的一部分
9.4.6 比较两个字符串
9.4.7 在一个字符串内搜索
9.4.8 大小写转换
9.4.9 数值转换
9.4.10 效率和strlib.h库
小结

复习题
程序设计练习
第10章 模块化开发
10.1 pig latin—一个模块化开发的 案例研究
10.1.1 应用自顶向下的设计
10.1.2 使用伪代码
10.1.3 实现translateline
10.1.4 考虑空格和标点符号的问题
10.1.5 精化单词的定义
10.1.6 设计记号扫描器
10.1.7 完成translateline的实现
10.1.8 定义扫描器模块接口
10.2 在模块中维护内部状态
10.2.1 全局变量
10.2.2 使用全局变量的危险性
10.2.3 保持变量的模块私有化
10.2.4 初始化全局变量
10.2.5 私有函数
10.3 实现扫描器抽象
小结
复习题
程序设计练习
第三部分 复合数据类型
第11章 数组
11.1 数组
11.1.1 数组声明
11.1.2 数组选择
11.1.3 一个简单的数组实例
11.1.4 改变下标值的范围
11.2 数据的内部表示法
11.2.1 比特、字节和字
11.2.2 内存地址
11.2.3 运算符sizeof
11.2.4 变量的内存分配
11.2.5 引用超出数组范围的元素
11.3 数组作为参数进行传递
11.3.1 元素个数的通用化
11.3.2 数组参数传递机制
11.3.3 实现函数printintegerarray和getintegerarray
11.3.4 实现函数reverseintegerarray
11.3.5 实现函数swapintegerelements
11.4 使用数组制作表格
11.5 数组的静态初始化
11.5.1 自动确定数组大小
11.5.2 确定初始化数组的大小
11.5.3 初始化数组和标量类型
11.6 多维数组

11.6.1 向函数传送多维数组
11.6.2 初始化多维数组
小结
复习题
程序设计练习
第12章 查找和排序
12.1 查找
12.1.1 在整数数组中查找
12.1.2 关于查找的另一个更复杂的例子
12.1.3 线性查找
12.1.4 二分查找
12.1.5 查找算法的相对效率
12.2 排序
12.2.1 对一个整数数组排序
12.2.2 选择排序算法
12.2.3 选择排序效率的评估
12.2.4 测试程序的运行时间
12.2.5 选择排序的算法分析
小结
复习题
程序设计练习
第13章 指针
13.1 将地址作为数据值
13.2 c语言的指针操作
13.2.1 在c语言中声明指针变量
13.2.2 基本的指针操作
13.2.3 特殊指针null
13.3 通过引用传递参数
13.3.1 设计函数swapinteger
13.3.2 用引用调用返回多个结果
13.3.3 过度使用引用调用的危险
13.4 指针和数组
13.4.1 指针运算
13.4.2 运算符＋＋和－－的新作用
13.4.3 指针的自增和自减
13.4.4 指针和数组的关系
13.5 动态分配
13.5.1 void *类型
13.5.2 动态数组
13.5.3 查找malloc中的错误
13.5.4 释放内存
小结
复习题
程序设计练习
第14章 再论字符串
14.1 string类型的概念表示
14.1.1 字符串作为数组

14.1.2 字符串作为指针
14.1.3 字符串作为抽象类型
14.1.4 字符串参数
14.1.5 字符串变量
14.1.6 指针和数组变量间的区别
14.1.7 决定字符串的表示方法
14.2 ansi字符串库
14.2.1 strcpy函数
14.2.2 strncpy函数
14.2.3 strcat和strncat函数
14.2.4 strlen、strcmp和strncmp函数
14.2.5 strchr、strrchr和strstr函数
14.2.6 ansi字符串函数的应用
14.3 strlib库的实现
14.3.1 实现转换函数
14.3.2 strlib分配函数的实现
小结
复习题
程序设计练习
第15章 文件
15.1 文本文件
15.2 c语言中文件的使用
15.2.1 声明file*类型的变量
15.2.2 打开文件
15.2.3 执行i/o操作
15.2.4 关闭文件
15.2.5 标准文件
15.3 字符i/o
15.3.1 文件更新
15.3.2 在输入文件中重新读取字符
15.4 面向行的i/o
15.5 格式化i/o
15.5.1 printf的三种形式
15.5.2 scanf函数
15.5.3 用scanf读入字符串
15.5.4 格式化i/o的一个实例
15.5.5 使用scanf的局限
小结
复习题
程序设计练习
第16章 记录
16.1 数据记录的概念
16.2 记录在c语言中的使用
16.2.1 定义一个结构类型
16.2.2 声明结构变量
16.2.3 记录选择
16.2.4 记录初始化

16.2.5 简单记录
16.3 数组与记录的结合
16.4 记录的指针
16.4.1 定义一个指向记录类型的指针
16.4.2 为记录数据分配空间
16.4.3 对记录指针进行操作
16.5 创建记录的数据库
16.5.1 创建员工信息数据库
16.5.2 数据库的使用
16.6 基于记录的应用程序设计
16.6.1 使用数据库的重要性
16.6.2 问题的框架
16.6.3 设计内部表示
16.6.4 设计外部结构
16.6.5 程序代码
16.6.6 数据驱动设计方法的重要性
小结
复习题
程序设计练习
第17章 深入学习
17.1 递归
17.1.1 递归的简单说明
17.1.2 factorial函数
17.1.3 递归信任
17.1.4 递归范例
17.1.5 排列的生成
17.1.6 用递归的思想思考
17.2 抽象数据类型
17.2.1 队列抽象
17.2.2 以队列抽象表示类型
17.2.3 queue.h 接口
17.2.4 实现队列抽象
17.2.5 队列抽象的另一种实现方法
17.3 算法分析
17.3.1 评估算法效率
17.3.2 o标记
17.3.3 再看选择排序
17.3.4 分而治之策略
17.3.5 合并两个数组
17.3.6 合并排序算法
17.3.7 合并排序的计算复杂性
17.3.8 比较平方复杂性与nlogn复杂性的性能
小结
复习题
程序设计练习
附 录
附录a c语言的语法和结构总结

附录b 库源代码

章节摘录

版权页：插图：用户的输入用加黑的字体表示，以便与程序所产生的输出结果相区别。
为了让大家更清楚用户的动作，图中还用符号来表示用户已经按下了Return或Enter键，这意味着输入行结束，而实际上这个符号并不会显示在屏幕上。
1.4 算法现在你已经对计算机系统的基本结构有所了解，下面我们介绍计算机科学。
由于计算机科学是在计算机的帮助下解决问题的学科，所以你应该了解算法（algorithm）e的概念，这个概念无论对计算机科学还是对解决问题的抽象学科来说都是基础。
通俗地说，算法是一种解决问题的策略，然而，为了理解计算机科学家如何使用这个术语，就必须将这种直观的理解正式化并给出严格定义。
要成为一个算法，解决问题的技术必须满足三个基本要求。
首先，算法必须用清楚的、明确的形式来表达，以使人们能够理解其中的每一个步骤。
第二，算法中的每一个步骤必须有效，以便人们在实践中能够执行它们。
例如，若某一算法包含“ 用兀的确切值与r相乘 ”这样的操作，则这个技术就不是有效的，因为无法算出兀的确切值。
第三，算法不能无休止地运行下去，而必须在有限的时间内给出一个答案。
简而言之，算法必须是：1）清楚、明确地定义。
2）有效，即每一步骤都切实可行。
3）有限，即可在有限步骤后得到结果。
在接下来学习复杂算法时，这些性质将变得更为重要。
现在，只要知道算法是种抽象的解决问题的策略，这种策略最终将成为你所编写的程序的核心就可以了。
慢慢地你会发现，与所要解决的问题一样，算法在复杂性上差别很大。
有些问题非常简单，很快就可以想到相应的算法，可以不费吹灰之力就编写一个解决这样问题的程序。
而当问题变得很复杂时，就需要更多的思考才能想出解决它的算法。
大多数情况下，解决一个问题可以使用几个不同的算法，在编写最终程序之前需要考虑许多潜在的解决方案。
第6章将仔细讨论这个问题，讨论如何决定某个特定问题的最佳算法。
1.5 程序设计语言和编译用计算机解决问题包括两个概念上不同的步骤。
首先，应该构造出一个算法或在解决该问题的已有算法中挑选一个。
这个过程称为算法设计（algorithm icdesign）。
第二步是用程序设计语言将该算法表达为程序，这个过程称为编码（coding）。
刚开始学习程序设计时，将算法翻译成实际的c语言的编码过程会是整个过程中较为困难的阶段。
作为一个新程序员，要从简单的问题入手，就像学习任何新技术时所做的那样。
简单的问题常有简单的解决方法，它的算法设计阶段不会有太大的挑战性。

编辑推荐

《C语言的科学和艺术》是C语言经典译丛之一。

<<C语言的科学和艺术>>

版权说明