



普通高等教育“十一五”国家级规划教材配套用书
21世纪高等院校计算机教材

数据结构习题解答 与实验指导 (第二版)

石 强 罗文劼 王 苗 编著

本书是作者积指导学生实验教学实践经验编写而成
帮助学生深入学习, 掌握并灵活运用数据结构知识



加深理解 强化操作



中国铁道出版社
CHINA RAILWAY PUBLISHING HOUSE



普通高等教育“十一五”国家级规划教材配套用书
21 世纪高等院校计算机教材

数据结构习题解答与实验指导 (第二版)

石 强 罗文劼 王 苗 编著

责任编辑 李 强

封面设计 李 强
中国铁道出版社
CHINA RAILWAY PUBLISHING HOUSE

内 容 简 介

本书是作者积多年讲授“数据结构”课程及指导学生实验的教学实践经验编写而成。作者力图通过指导学生的实验实践和大量典型例题的解析,帮助学生深入学习、掌握并灵活运用数据结构知识。

全书由两部分组成。第一部分为学习提要和习题解答,其内容为数据结构各部分的知识要点、典型例题解析和习题解答,可以帮助学生提纲挈领地掌握知识重点、巩固所学内容;第二部分为实验指导,根据数据结构课程的教学重点,给出7个实验题目,每个题目都有明确的实验要求,同时还对实验步骤和实验报告进行规范,还提供了一个实验报告范例。

本书作为普通高等教育“十一五”国家级规划教材《数据结构(第二版)》的配套教材,可以配合主教材使用,起到衔接课堂教学与实验教学、课下辅导的作用,可作为高等院校学生学习“数据结构”课程的参考书,也可作为研究生入学考试的辅导材料,对于从事计算机应用及开发的技术人员以及广大的计算机及相关专业的自学者,也具有一定的参考价值。

图书在版编目(CIP)数据

数据结构习题解答与实验指导 / 石强, 罗文劫, 王苗
编著. —北京: 中国铁道出版社, 2007. 11
21 世纪高等院校计算机教材
ISBN 978-7-113-07756-3

I. 数… II. ①石…②罗…③王… III. 数据结构—高等学校—教学参考资料 IV. TP311.12

中国版本图书馆 CIP 数据核字 (2007) 第 180312 号

书 名: 数据结构习题解答与实验指导 (第二版)

作 者: 石 强 罗文劫 王 苗

出版发行: 中国铁道出版社 (100054, 北京市宣武区右安门西街 8 号)

策划编辑: 严晓舟 秦绪好

责任编辑: 翟玉峰 包 宁

封面制作: 白 雪

印 刷: 三河市宏达印刷有限公司

开 本: 787×1092 1/16 印张: 12.25 字数: 281 千

版 本: 2007 年 12 月第 2 版 2007 年 12 月第 1 次印刷

印 数: 1~5 000 册

书 号: ISBN 978-7-113-07756-3/TP·2121

定 价: 19.00 元

版权所有 侵权必究

本书封面贴有中国铁道出版社激光防伪标签, 无标签者不得销售

凡购买铁道版的图书, 如有缺页、倒页、脱页者, 请与本社计算机图书批销部调换。

21 世纪高等院校计算机教材

编写委员会

主 任：王凤先

副主任：马胜甫 刘振鹏

委 员：（按姓氏笔画排列）

于 锋	王学军	王卫华	王春红	白乙拉
刘立民	刘清波	李亚平	李继民	张小莉
杨怀卿	孟玉芹	赵英杰	赵新生	赵福来
徐建民	韩宪忠			

第二版前言

数据结构是计算机科学与技术等电子信息类相关专业的一门重要的基础课程。通过对数据结构相关知识的理论学习,学生可以较全面地理解算法和数据结构的概念,掌握各种数据结构和算法的实现方式,比较不同的数据结构和算法的特点。数据结构是一门理论与实践紧密结合的课程,不仅要能够理解基础理论知识,针对具体问题选择和设计出适当的逻辑结构、存储结构及相应的算法,还要能在此基础上编写出结构清晰、正确易读、符合软件工程规范的程序,从而为进一步学习后继专业课程和软件开发打下良好的基础。

为了帮助读者提高求解数据结构习题的能力,作者结合多年讲授数据结构课程的经验编写了本书的理论知识与习题解答部分。将各章的知识要点进行了归纳总结,对重点和难点进行了再次阐述,精心选择了许多典型例题进行了解析,并且对配套教材上的课后习题给予了详细的解答。

本书根据国内使用计算机的情况,在内容上力图具有一定的先进性和较大的适应性。遵循这一原则,在编写中着重讲述原理、概念和实例。为了帮助读者提高应用数据结构、通过程序设计解决实际应用问题的能力,作者编写了实验指导部分。根据数据结构课程内容的需要,给出了7个实验题目,对每个题目给出了明确的实验要求,同时还给出了规范的实验步骤和实验报告范例。

本书与入选普通高等教育“十一五”国家级规划教材的《数据结构(第二版)》相配套。在第一版的基础上按照第二版教材进行了修改。本书保持了前一版的基本框架,进一步完善了算法,增加和改善了重点算法的注释。第一部分由罗文劼、王苗修订,第二部分由石强修订,全书由石强统稿。

本书分为两部分:第一部分是理论知识与习题解答,第二部分是实验指导。第一部分的内容与《数据结构(第二版)》一书相对应,也分为10章,每一章都由内容概述、重点难点指导、典型例题解析、课后习题选解等部分组成;第二部分的内容根据数据结构课程的重点,给出了7个实验题目,每个实验题目采用了统一的格式,由问题描述、数据结构设计、功能(函数)设计、界面设计、编码实现、运行与测试几个部分组成,为学生提出了明确的实验要求,并对实验步骤给予指导。

本书在写作和修订过程中,参考了一些国内外教材及数据结构习题集和辅导书,并且从互联网中汲取了不少数据结构方面的精华题目,得到了许多专家和众多院校数据结构任课教师的大力支持和帮助,他们提出了许多中肯的意见和很好的建议,对本书的修订起到了很大的指导作用。对此作者表示衷心的感谢。

在本书的编写过程中,刘振鹏、张小莉等老师对本书的编写提出了许多宝贵意见,并给予了大力支持,在此表示诚挚谢意。感谢作者的多位同事和学生,许百成、史青宣、苗秀芬、王硕等老师和许多学生对本书的内容提出了很多的修改意见,在使用本书的过程中指出了书中的一些不足,使得本书更为完善。

由于编者水平有限,书中难免有不妥之处,恳请读者批评指正,编者不胜感激。

编者

2007年11月

第一版前言

数据结构是计算机专业的一门核心课程。在应用计算机解决实际问题时，首先要解决的就是如何将问题以计算机能接受的形式表示出来，然后再将解决问题的方法步骤用计算机能识别的形式告诉计算机，让计算机自动执行求解。这一过程就是人们所说的程序设计过程。

“数据结构”课程的内容正是在长期的程序设计实践中提炼、升华而成的，主要研究信息的逻辑结构及其基本操作在计算机中的表示和实现，是计算机科学的算法理论基础和软件设计的技术基础。在计算机专业中数据结构是操作系统、数据库原理、编译原理等后续课程的基础，在计算机专业课程的学习中起着承上启下的作用。

由于数据结构具有概念性强，内容灵活，所涉及数据的组织、存储以及操作的方法比较抽象等特点，因此，对于初学者来说，往往找不到感觉，不知道如何学习这门课，面对习题更无从下手。作者想借编写本书的机会结合多年讲授本门课程的经验，将各章的知识要点进行归纳和总结，对难以理解的问题进行通俗的讲解和指导，对涉及到重要知识点的典型题目进行分析解答，目的是帮助读者理解数据结构的内容，尽快掌握各种数据结构的表示方法及应用实现，同时求解数据结构习题的能力也能有一个明显的提高。

此外，数据结构还是一门实践性很强的课程，只是看书本、做习题是绝对不够的。因此，在数据结构的教学中，除了课堂教学外，每周还应有不少于两个学时的实验课。在数据结构的课程实验中要解决的问题更接近于实际，不同于平时的编写功能单一的“小”算法的练习，实验是软件设计的综合训练，包括问题分析、总体结构设计、用户界面设计、程序设计基本技能和技巧，多人合作，以至一整套软件工作规范的训练和科学作风的培养。为此，作者根据数据结构课程内容的需要，给出7个实验题目，并对每个题目提出明确的实验要求，同时还对实验步骤和实验报告进行了规范。

本书与中国铁道出版社出版的《数据结构》教材相配套，主要内容由两部分组成：理论知识与习题解答部分和实验题目与指导部分。其中理论知识与习题解答部分与《数据结构》一书相对应，也分10章，每一章都由内容概述、重点难点指导、典型例题解析以及课后习题选解等部分组成；实验题目与指导部分根据数据结构课程的教学重点，给出7个实验题目，每个实验题目采取了统一的格式，由问题描述、数据结构设计、功能（函数）设计、界面设计、编码实现、运行与测试几个部分组成，为学生提出明确的实验要求，并对实验步骤给予指导。

在本书的编写过程中，参考了一些国内外优秀教材及数据结构习题集和辅导书。刘振鹏、张小莉等老师对本书的编写提出了许多宝贵意见，并给予大力支持，在此表示诚挚谢意。

本书在编写过程中力求概念清晰，表述正确，通俗易懂，便于自学。希望读者通过对本书的学习，能够更全面、更透彻地理解和掌握数据结构这门课程。但由于编者水平有限，书中难免存在疏漏或不妥之处，恳请读者批评指正，编者不胜感激。

编者

2004年10月

目 录

第一部分 习题解答

第 1 章 绪论	1
1-1 重点难点指导	1
1-1-1 相关术语	1
1-1-2 算法的描述和分析	3
1-2 典型例题解析	3
1-2-1 选择题	3
1-2-2 判断题	4
1-2-3 填空题	5
1-3 课后习题选解	5
第 2 章 线性表	8
2-1 重点难点指导	8
2-1-1 相关术语	8
2-1-2 线性表的顺序存储	9
2-1-3 链表	10
2-1-4 线性表的基本运算	12
2-2 典型例题解析	15
2-2-1 选择题	15
2-2-2 判断题	16
2-2-3 简答题	17
2-2-4 算法设计题	18
2-3 课后习题选解	21
第 3 章 栈和队列	26
3-1 重点难点指导	26
3-1-1 相关术语	26
3-1-2 栈	27
3-1-3 队列	29
3-1-4 栈的应用	33
3-1-5 队列的应用	33
3-2 典型例题解析	33
3-2-1 选择题	33
3-2-2 填空题	35
3-2-3 简答题	35

3-2-4 算法设计题	36
3-3 课后习题选解	40
第 4 章 串	43
4-1 重点难点指导	43
4-1-1 相关术语	43
4-1-2 串的基本运算	43
4-1-3 串的存储结构	44
4-1-4 字符串匹配算法实现	46
4-2 典型例题解析	48
4-2-1 字符串的基本运算题	48
4-2-2 算法设计题	49
4-3 课后习题选解	51
第 5 章 数组和广义表	54
5-1 重点难点指导	54
5-1-1 相关术语	54
5-1-2 多维数组	55
5-1-3 特殊矩阵	55
5-1-4 稀疏矩阵	57
5-1-5 广义表	58
5-2 典型例题解析	59
5-2-1 填空题	59
5-2-2 简答题	60
5-2-3 算法设计题	61
5-3 课后习题选解	63
第 6 章 二叉树	70
6-1 重点难点指导	70
6-1-1 相关术语	70
6-1-2 二叉树的存储结构	71
6-1-3 二叉树的遍历运算	72
6-1-4 二叉树的基本应用	74
6-2 典型例题解析	74
6-2-1 选择题	74
6-2-2 判断题	75
6-2-3 填空题	76
6-2-4 应用题	77
6-2-5 算法设计题	78
6-3 课后习题选解	83

第 7 章 树和森林	88
7-1 重点难点指导	88
7-1-1 相关术语	88
7-1-2 树的存储结构	89
7-1-3 树的基本运算	90
7-1-4 树、森林和二叉树的相互转换	91
7-2 典型例题解析	91
7-2-1 选择题	91
7-2-2 判断题	92
7-2-3 填空题	92
7-2-4 应用题	93
7-2-5 算法设计题	93
7-3 课后习题选解	94
第 8 章 图	98
8-1 重点难点指导	98
8-1-1 相关术语	98
8-1-2 图的基本操作	100
8-1-3 图的存储表示	100
8-1-4 图的遍历	103
8-1-5 图的应用	104
8-2 典型例题解析	107
8-2-1 判断题	107
8-2-2 选择题	108
8-2-3 填空题	109
8-2-4 应用题	109
8-2-5 算法设计题	111
8-3 课后习题选解	116
第 9 章 查找	124
9-1 重点难点指导	124
9-1-1 相关术语	124
9-1-2 线性表查找	124
9-1-3 排序树上的查找	126
9-1-4 哈希表	128
9-2 典型例题解析	130
9-2-1 判断题	130
9-2-2 选择题	130
9-2-3 应用题	131

88	9-2-4 算法设计题	132
88	9-3 课后习题选解	135
第 10 章 排序		143
88	10-1 重点难点指导	143
00	10-1-1 相关术语	143
10	10-1-2 插入排序	144
10	10-1-3 交换排序	145
10	10-1-4 选择排序	146
50	10-1-5 归并排序	147
50	10-1-6 基数排序	147
50	10-1-7 外部排序	148
50	10-2 典型例题解析	148
40	10-2-1 判断题	148
80	10-2-2 选择题	149
80	10-2-3 填空题	150
80	10-2-4 算法设计题	151
001	10-3 课后习题选解	154
第二部分 实验指导		
第 11 章 实验的一般步骤		160
501	11-1 概述	160
501	11-2 实验步骤	161
第 12 章 实验安排		163
901	实验一 Josephus 环问题	163
901	实验二 一元多项式相加问题	165
111	实验三 停车场模拟管理程序的设计与实现	167
911	实验四 农夫过河问题的求解	171
451	实验五 简单哈夫曼编/译码的设计与实现	173
451	实验六 简单校园导游程序的设计与实现	175
451	实验七 简单个人书籍管理系统的设计与实现	177
参考文献		180
附录 A 实验报告范例		181

第 1 章 绪 论

本章主要介绍一些关于数据结构的概念、学习数据结构的意义，以及描述算法和分析算法的方法。

重点提示：

- 有关数据结构的术语及概念
- 理解数据结构的内容
- 理解逻辑结构、存储结构和数据运算 3 方面的概念及相互之间的关系
- 掌握估算时间复杂度和空间复杂度的度量方法

1-1 重点难点指导

1-1-1 相关术语

1. 数据、数据元素、数据项和数据对象

数据：是信息的载体，它能够被计算机识别、存储和加工处理。它是计算机程序加工的原料，应用程序处理各种各样的数据。它可以是数值数据，也可以是非数值数据。数值数据是一些整数、实数或复数，主要用于工程计算、科学计算和商务处理等；非数值数据包括字符、文字、图形、图像、语音等。

数据项：数据项是具有独立含义的最小单位。有些数据元素是由若干个数据项组成的。数据项有名和值之分，数据项名是一个数据项的标识，用变量定义，而数据项值是它的一个可能取值。数据项具有一定的类型，依数据项的取值类型而定。

数据元素：是数据的基本单位。在不同的条件下，数据元素又可称为元素、结点、顶点、记录等。一个数据元素可由若干个数据项组成，这些数据项可以分为两种：一种叫做初等项，这些数据项是在数据处理时不能再分割的最小单位；另一种叫做组合项。

数据对象或数据元素类：是具有相同性质的数据元素的集合。在某个具体问题中，数据元素都具有相同的性质（元素值不一定相等），属于同一数据对象（数据元素类），数据元素是数据元素类的一个实例。

2. 数据类型

数据类型是一个值的集合以及在这些值上定义的一组操作的集合。

在高级程序设计语言中，数据类型可分为如下两类。

- 原子类型：其值不可分解。如 C 语言中的整型、实型、字符型等。
- 结构类型：其值可分解为若干成分。如 C 语言中的数组、结构等。

3. 抽象数据类型

抽象数据类型（Abstract Data Type, ADT）是指抽象数据的组织和与之相关的操作。它可以看作是数据的逻辑结构及其在逻辑结构上定义的操作。

抽象数据类型的特征是使用与实现相分离, 实行封装和信息隐蔽。也就是说, 在抽象数据类型设计时, 把类型的定义与其实现分离开来。

4. 数据结构

数据结构是指互相之间存在着一种或多种关系的数据元素的集合, 是指数据元素之间的相互关系, 即数据的组织形式, 它包括以下 3 方面的内容:

- 逻辑结构: 数据之间的逻辑关系。
- 存储结构: 数据元素及其关系在计算机存储器内的表示。
- 数据的运算: 即对数据对象施加的操作。

5. 两类逻辑结构

(1) 线性结构

线性结构的逻辑特点是: 若结构为非空集, 有且仅有一个开始结点和一个终端结点, 并且所有结点都最多只有一个直接前驱和直接后继, 如线性表。

要点: 只有一个直接前驱和直接后继。

(2) 非线性结构

非线性结构的逻辑特点是: 一个结点可能有多个直接前驱和直接后继。如树形结构和图形结构。

要点: 可能有多个直接前驱和多个直接后继。

6. 数据逻辑结构的 4 种基本形态

- 集合结构: 数据元素间的关系是“属于同一个集合”。
- 线性结构: 数据元素之间存在着一对一的关系。
- 树形结构: 数据元素之间存在着一对多的关系。
- 图形结构: 数据元素之间存在着多对多的关系。

7. 4 种常见的存储结构

(1) 顺序存储

顺序存储方法是把逻辑上相邻的元素存储在物理位置相邻的存储单元中, 由此得到的存储表示称为顺序存储结构。顺序存储结构是一种最基本的存储表示方法, 通常借助于程序设计语言中的数组来实现。

要点: 物理位置相邻的存储单元。

(2) 链式存储

链式存储方法对逻辑上相邻的元素不要求其物理位置相邻, 元素间的逻辑关系通过附设的指针字段来表示, 由此得到的存储表示称为链式存储结构, 链式存储结构通常借助于程序设计语言中的指针类型来实现。

要点: 不要求物理位置相邻的存储单元, 元素间的逻辑关系通过附设的指针域来表示。

(3) 索引存储方式

要点: 是通过建立索引表存储结点信息的方法, 其中索引表一般存储结点关键字和一个地点信息, 可通过该地址找到结点的其他信息。有稠密索引和稀疏索引之分。

(4) 散列存储方式

要点: 是根据结点的关键字直接计算出该结点的存储地址的方法。

1-1-2 算法的描述和分析

1. 算法

算法是对特定问题求解步骤的一种描述,是指令的有限序列。其中每一条指令表示一个或多个操作。一个算法应该具有下列特性:

- 有穷性。一个算法必须在有穷步之后结束,即必须在有限时间内完成。
- 确定性。算法的每一步必须有确切的定义,无二义性。算法的执行对应着的相同的输入仅有唯一的一条执行路径。
- 可行性。算法中的每一步都可以通过已经实现的基本运算的有限次执行得以实现。
- 输入。一个算法具有零个或多个输入,这些输入取自特定的数据对象集合。
- 输出。一个算法具有一个或多个输出,这些输出同输入之间存在某种特定的关系。

2. 对算法的设计要求

- 正确。算法的执行结果应当满足预先规定的功能和性能要求。
- 可读。一个算法应当思路清晰、层次分明、简单明了、易读易懂。
- 健壮。当输入不合法数据时,应能进行适当处理,不至于引起严重后果。
- 高效。有效地使用存储空间和有较高的时间效率。

3. 算法的性能分析与度量

(1) 时间复杂度

① 某算法的时间复杂度是执行该算法所耗费的时间。通常某算法的时间复杂度是问题规模 n 的函数 $T(n)$ 。

② 大 O 记法:表示算法的渐进时间复杂度。

③ 常见的渐进时间复杂度有:

$$O(1) < O(\log_2 n) < O(n) < O(n \log_2 n) < O(n^2) < O(n^3) < O(2^n)$$

(2) 空间复杂度

一个算法的空间复杂度是指该算法所耗费的存储空间。它通常也是问题规模 n 的函数 $T(n)$ 。

1-2 典型例题解析

1-2-1 选择题

1. 数据结构是一门研究非数值计算程序设计中计算机的 ① 以及它们之间的 ② 和运算等的学科。

- ① A. 操作对象 B. 计算方法 C. 逻辑存储 D. 数据映像
② A. 结构 B. 关系 C. 运算 D. 算法

【分析】数据结构是一门研究非数值计算程序设计中计算机的操作对象以及它们之间的关系和运算等的学科。

【答案】① A ② B

2. 从逻辑结构上可以把数据结构分为_____。

- A. 动态结构和静态结构 B. 紧凑结构和非紧凑结构
C. 线性结构和非线性结构 D. 内部结构和外部结构

【分析】数据的逻辑结构有两类, 分别是线性结构和非线性结构。

【答案】C

3. 线性结构的顺序存储结构是一种_____①_____的存储结构, 线性结构的链式存储是一种_____②_____的存储结构。

- A. 随机存取 B. 顺序存取 C. 索引存取 D. 散列存取

【分析】顺序存储结构是一种随机存取结构, 链式存储结构是一种顺序 (一个个结点的查找) 存取结构。

【答案】① A ② B

4. 下面程序的时间复杂度为_____。

```
for(i=0;i<m;i++)
    for(j=0;j<n;j++)
        A[i][j]=i*j;
```

- A. $O(m^2)$ B. $O(n^2)$ C. $O(m \times n)$ D. $O(m+n)$

【分析】第一个 for 循环执行 m 次, 第二个 for 循环执行 n 次, 两个 for 循环嵌套起来共执行 $m \times n$ 次。

【答案】C

1-2-2 判断题

1. 顺序存储方式只能用于线性结构, 不能用于非线性结构。

【分析】顺序存储方式能用于存储非线性结构。如存储树形结构中, 完全二叉树的数组存储和堆排序中堆的存储。

【答案】错误

2. 基于某种逻辑结构之上的运算, 其实现是唯一的。

【分析】基于某种逻辑结构, 其存储结构不唯一, 因此运算也就不唯一。

【答案】错误

3. 数据元素是数据的最小单位。

【分析】数据元素是数据的基本单位, 数据元素可以由数据项组成, 且数据项是数据的最小单位。

【答案】错误

4. 算法可以用不同的语言描述, 如果用 C 语言或 PASCAL 等高级语言来描述, 则算法实际上就是程序。

【分析】算法用各种计算机语言描述则表现为一个程序但并不等于程序, 因为程序的逻辑不一定能满足有穷性。

【答案】错误

5. 数据结构是带有结构的数据元素的集合。

【分析】若把数据结构进行形式化描述, 则可从逻辑上认为数据结构 DS 是数据元素的集合 D 和 D 上关系的集合 R 所构成的二元组: $DS=(D,R)$, 这里的关系 R 用于描述数据之间的逻辑关系, 即数据的结构。

【答案】正确

1-2-3 填空题

1. 线性结构中元素的关系是_____, 树形结构中元素的关系是_____, 图形结构中元素的关系是_____。

【答案】一对一; 一对多; 多对多

2. 算法的 5 个重要特性是_____, _____, _____, _____和_____。

【答案】有穷性; 确定性; 可行性; 输入; 输出

1-3 课后习题选解

1. 解释下列术语: 数据、数据元素、数据对象、数据结构、存储结构、线性结构、算法、数据类型。

参见教材。

2. 说明数据结构的概念与程序设计语言中数据类型概念的区别和联系。

数据类型是一个值的集合以及在这些值上定义的一组运算的集合。如 C 语言中的 `int` 类型, 表示集合 $\{x|x \text{ 是整数且 } -32768 \leq x \leq 32767\}$, 其运算包括“+”、“-”、“×”、“/”和“%”等。

数据结构指存在特定关系的数据元素的集合。数据结构包括 3 方面的内容: ①数据的逻辑结构, 指数据元素之间的逻辑关系; ②数据的存储结构, 指数据元素及其关系在计算机中的实现方法; ③数据的运算, 指对数据元素施加的操作。

二者都是数据元素和运算的集合。但是数据结构还涉及数据元素之间的逻辑关系。

3. 讨论顺序存储结构和链式存储结构各自的特点、适用范围, 并说明在实际应用中应如何选取数据存储结构。

① 顺序存储结构用存储位置的相邻表示逻辑关系的相邻。需要占用连续的存储单元。可以随机访问数据元素。

② 链式存储结构用附加的指针表示逻辑关系。不需要占用连续的存储空间。只能顺序地访问数据元素。

③ 顺序存储结构是静态分配内存的方式, 链式存储结构是动态分配内存的方式, 它在需要时分配存储单元, 不需要时释放存储单元。当对于数据元素需要频繁地插入、删除时应采用链式存储结构, 反之, 采用顺序存储结构。

4. 试写一算法, 从大到小依次输出顺序读入的 3 个整数 x 、 y 和 z 的值。

【提示】通过比较和交换使得 3 个数从大到小有序。

```
main() {
    int x,y,z,t;
    cin>>x;
```

```

cin>>y;
cin>>z;           //读入 3 个整数 x、y 和 z
if (x<y){
    t=x;
    x=y;
    y=t;
}
if (x<z){
    t=x;
    x=z;
    z=t;
}
if (y<z){
    t=y;
    y=z;
    z=t;
}
cout<<x<<y<<z<<endl; //输出从大到小有序的 3 个数
}
    
```

5. 已知 k 阶裴波纳契序列的定义为

$$f_0=0, f_1=0, \dots, f_{k-2}=0, f_{k-1}=1;$$

$$f_n=f_{n-1}+f_{n-2}+\dots+f_{n-k}, n=k, k+1, \dots$$

试编写求 k 阶裴波纳契序列的第 m 项值的函数算法, k 和 m 均以参数的形式在参量表中出现。

【提示】采用递归的方法实现。

```

int fun(int k, int m){
    //求 k 阶裴波纳契序列的第 m 项值
    int f;
    if (m<k-1)
        f=0;
    else
        if (m==k-1)
            f=1;
        else{
            f=0;
            for (j=1; j<=k; j++)
                f=f+fun(k, m-j);
        }
    return f;
}
    
```

6. 编写一个程序计算一元多项式 $P_n(x)=P_0+P_1x+P_2x^2+\dots+P_nx^n$ 的值 $P_n(x_0)$, 设 n, x_0 和 P_i ($0 \leq i \leq n$) 均为已知量, 从键盘读入。程序中 P_i ($0 \leq i \leq n$) 的值是采用什么结构存储的?

【提示】将一元多项式改写为如下形式:

$$P_n(x)=P_0+P_1x+P_2x^2+\dots+P_nx^n=((\dots((P_nx+P_{n-1})x+P_{n-2})x+\dots+P_1)x+P_0$$

```

main(){
    int n, i;
    double x, p, s;
    cin>>n;
    
```

```

cin>>x;           //读入 n, x 的值
s=0;
for(i=0;i<=n;i++){
    cin>>p;
    s=s*x+p;
}
cout<<s<<endl;
}

```

P_i 的读入次序为: $P_n, P_{n-1}, \dots, P_1, P_0$

7. 试编写算法计算 $i! \times 2^i$ 的值并存入数组 $a[\text{arrsize}]$ 的第 i 个分量中 ($i=1, 2, \dots, n$)。假设计算机中允许的整数最大值为 maxint , 则当 $n > \text{arrsize}$ 或对某个 k ($1 \leq k \leq n$) 使 $k! \times 2^k > \text{maxint}$ 时, 应按出错处理。可有下列 3 种不同的处理方式。

- (1) 用 ERROR 语句终止执行并报告错误。
 - (2) 用返回值 0 或 1 实现算法以区别正确返回或错误返回。
 - (3) 在参数表中设置一整型变量以区别正确返回或某种错误返回。
- 试讨论这 3 种方法各自的优缺点, 并以你认为较好的方式实现。

【提示】采用第 2 种处理方式。

```

#define arrsize ...
#define Maxint ...
int fun1(int i){
    int j,s;           //计算 i!
    j=1;
    s=1;
    for(j=1;j<=i;j++){
        s=s*j;
    }
    return s;
}
int fun2(int i){
    int j,s;           //计算 2^i
    j=1;
    s=1;
    for(j=1;j<=i;j++){
        s=s*2;
    }
    return s;
}
int a[arrsize+1]
main(){
    int i;
    for(i=1;i<=arrsize;i++){
        a[i]=fun1(i)*fun2(i); //计算 i!*2^i
        if(a[i]>Maxint)
            break;
    }
    if(i<arrsize)
        return 0;
    else
        return 1;
}

```