数据结构教学大纲

2017 年 9 月

目录

第一部分　大纲说明

1.1 制定依据

本教学大纲是依据河北师范大学软件学院 2016 年数据结构课程教学大纲和 16 级软件工程专业的教学计划进行的修订。本教学大纲的修订同时参考了 2016～2017 学年第一学期河北师范大学软件学院 15 级数据结构课程教学活动和教学效果。

1.2 适用范围

本教学大纲适用于河北师范大学软件学院软件工程专业的本科生教学。

1.3 课程性质

数据结构是软件工程专业的一门核心专业基础课。课程从 ADT 思想出发，介绍了三大类数据结构（线型、树型和图型）、两大类基本算法（查找和排序）以及算法分析的基础。本课程的开设位于大学二年级的第一学期，学生在完成离散数学、c 语言程序设计课程之后通过本课程的学习，进一步提升分析问题、解决问题的能力（分析能力、抽象能力、设计能力和编程能力等），并且通过对基本数据结构和常用算法的学习和积累，为今后进一步深入学习其他专业课程打下良好的基础。

1.4 教学目标

通过本课程的学习，要求学生达到以下基本目标：

1. 熟练掌握本课程的基本概念、基本原理、基本算法；熟练掌握用 C 语言实现本课程的基本数据结构和基本算法，并具备用 C 语言设计和实现有一定难度的算法的能力。熟练掌握和运用 C 语言的高级语法特性，包括：结构体、指针、函数指针、typedef、动态内存分配、可变参数函数和递归函数，等；
2. 掌握从分析问题到编写伪代码、编写程序、调试程序和测试程序的程序设计思路和方法；
3. 熟练掌握三大类数据结构（线型、树型和图型）之不同物理结构（顺序和链式）的 ADT 实现，即，能够根据给定的 ADT 规格说明熟练的编写该 ADT 实现的代码。ADT

实现的方式包括：C 实现、C++类实现和 STL 实现，本课程只要求用 C 语言实现；

4. 了解三大类数据结构能够解决的典型问题，能够对这些典型问题进行编程求解；
5. 熟练掌握各种查找算法；
6. 熟练掌握各种内部排序算法；
7. 熟练掌握各种数据结构中涉及重要性质、重要定理（及其证明过程）和相关的计算；
8. 深入理解递归程序的工作原理，熟练掌握递归程序的编写，并且能够把递归程序修改成非递归程序；
9. 理解和掌握算法分析的方法和原理，能够计算给定算法的时间复杂度和空间复杂度；
10. 能够分析和比较不同数据结构的优劣，进而会根据不同的问题选取适当的数据结构；
11. 对抽象数据类型 ADT 的核心思想能够有较深刻的理解，能够对实际问题在已有的数据结构基础上进行灵活的定制和剪裁。能够对实际问题中隐含的数据结构进行分析和识别，并能用 ADT 的思想进行程序设计；
12. 体会 ADT 与类模板的关系。

第二部分　教学设计

2.1 教学手段

教学手段分两种：理论教学和实践教学。理论教学在课上完成，采用多媒体教学手段，主要借助短小精悍的示例代码来介绍重要的概念、重要的思想和重要的方法。理论部分的教学采用课上教学和课下自学相结合的方式进行，课上讲解最基础和最重要的概念，其他内容由学生课下学习，培养学生的自学能力。实践教学包括两种形式：实验教学和课程设计。其中实验教学在课上进行，完成不了的部分学生可以利用课下时间来完成。实验教学要求学生在专业教师的指导和带领下根据实验手册中的实验要求，完成相应程序代码的编码、调试和测试，对理论教学中的方法和思想进行模仿和复现，达到强化编程技能，强化对重要概念、重要思想和重要方法的理解和掌握的目的。实验教学要求专业教师对学生就实验手册中的实验任务进行集中指导（一般为一节课的时间）以及个别辅导（一般为一节课的时间）。课程设计不占用课上时间，通过相对完整的开发需求，对课程中涉及的大多数数据结构进行综合的运用。

2.2 授课思路

1. 数据结构主体内容的讲授思路如下，分以下四个步骤：

(1)提出问题A → (2)解决问题 → (3)数据结构 → (4)解决问题

(1) 提出问题A

首先提出一个具有一定实际意义的问题A让学生编程解决。该问题应该具有一定的特质，比如：有趣、有一定的实用性、比较简单（代码量不大）、隐藏着一种或多种数据结构。

(2) 解决问题

学生通过编程来解决问题A，但此前先不介绍任何数据结构知识。

(3) 数据结构

通过(2)解决问题，引导学生观察和认识数据结构的存在（重点），并对数据结构进行一定程度的提炼和抽象（引出 ADT 主体）（重点），然后让学生实现这个 ADT（重点）。

(4) 解决问题

让学生用实现了的 ADT 函数库来再一次解决问题A或者去解决一个类似的或引申的问

题，让他们体验 ADT 的价值和意义。

2. 基本算法部分的授课思路：实例演示、程序演示、辅助教学软件演示相结合，力求做到简洁、直观和形象。

2.3 学时分配

本课程总学时为 72 学时，共 3 学分。其中理论教学 52 学时，实践教学（实验教学）20 学时，理论学时和实践学时的比例为 5:2。这两部分教学课时分配情况分别如下。

注：

1) 实践教学中的课程设计不分配课时，即不占用上课时间。

2) 实践教学中的实验教学如果课上完成不了，就以课下作业的方式由学生独立完成。

1. 理论课时

序号	教学内容	学时	备注
1	绪论	5	
2	线性表	6	
3	阶段小结	3	
4	栈和队列	4	
5	阶段小结	2	
6	串	5	
7	数组和广义表	6	
8	阶段小结	2	
9	树和二叉树	7	
10	图	8	
11	阶段小结	3	
12	查找	7	
13	内部排序	8	
14	总复习	3	
合计		69	

2. 实践课时（实验课时）

序号	教学内容	学时	备注
1	线性表	6	
2	栈和队列	6	
3	数组和广义表	2	
4	树和二叉树	3	
5	图	2	
6	内部排序	2	
合计		21	

2.4 课程考核

本课程的成绩由以下三部分组成：

1. 期中考试（教考分离）：20%
2. 平时成绩（出勤、书面作业和实践作业）：20%
3. 期末考试（教考分离）：60%

第三部分　目标细化

3.1 绪论

3.1.1 主要内容

1. 数据结构、逻辑结构、存储结构、数据类型和算法等基本概念。
2. 数据结构课程的起源和地位。
3. 抽象数据类型 ADT 的定义、表示和实现。
4. 类 C 语法的介绍。
5. 算法和算法分析工具。

3.1.2 教学目标

1. 熟悉各名词术语的含义，掌握基本概念，特别是数据的逻辑结构和存储结构之间的关系。
2. 了解数据结构在程序设计领域的重要地位和意义。
3. 掌握抽象数据类型 ADT 的定义、表示和实现。
4. 理解 ADT 的思想。
5. 理解算法五个要素的确切含义。
6. 掌握算法时间复杂度的计算。
7. 掌握算法空间复杂度的计算。

3.1.3 教学重点

1. 基本概念。
2. ADT 的定义、表示和实现方法。
3. ADT 的思想。
4. 算法分析，尤其是算法的时间复杂度分析。

3.1.4 教学难点

1. ADT 的定义、表示和实现。学生接触编程的时间较短，很少接触复杂的程序，不容易理解 ADT 的概念。ADT 的概念属于程序设计层面的问题，一般学生多少有一些程序编写的经验，缺乏程序设计的经验。
2. ADT 的思想。ADT 的思想属于软件工程的重用思想，学生没有接触软件工程的课程，也缺乏实际项目经验，因此理解 ADT 思想会比较困难。
3. 算法的时间复杂度分析。时间复杂度分析可能会涉及比较复杂的数学计算，尤其是算法中基本操作的频度计算可能需要借助一些数学定理或公式才能求解。另外，需要学生对程序的流程，尤其是多重循环和判断的嵌套比较清楚。

3.2 线性表

3.2.1 主要内容

1. 线性表的结构特征。
2. 线性表的逻辑结构定义、抽象数据类型定义和各种存储结构的描述方法。
3. 在线性表的两类存储结构（顺序的和链式的）上实现基本操作。
4. 线性表的一些复杂操作的实现，如：$A \cup B$、顺序合并、$(A-B) \cup (B-A)$，等。
5. 对线性表典型操作（或算法）的效率分析。
6. 循环链表和双向链表。

3.2.2 教学目标

1. 了解线性表的逻辑结构特性。
2. 熟悉各名词术语的含义，掌握基本概念。
3. 理解并熟练掌握线性表两类存储结构的 C 语言描述。
4. 理解并掌握 C 语言描述数据结构的方法和思路。
5. 熟练掌握线性表在两种存储结构上基本操作的算法实现。
6. 了解静态链表，加深对链表本质的理解。
7. 能够从时间和空间复杂度的角度综合比较线性表两种存储结构的不同特点及其适用场合。

3.2.3 教学重点

1. 线性表两类存储结构的 C 语言描述。
2. 线性表在两类存储结构上基本操作的算法实现。
3. 不同存储结构的特点、优缺点和适用场合。

3.2.4 教学难点

1. 线性表数据结构的 C 语言描述。学生们对 C 语言结构体语法掌握的程度以及对自定义数据类型的熟悉程度会直接影响这部分内容的掌握。
2. 静态链表和动态链表。静态链表的操作需要获取或修改数据元素存放的游标，以游标为数组下标进行数据元素的访问，游标和数组下标的关系以及之间的操作学生初次接触会有些困惑。学生对 C 语言指针语法掌握的程度以及对动态内存分配的熟悉程度会直接影响动态链表的算法实现。

3.3 栈和队列

3.3.1 主要内容

1. 栈和队列的结构特性。
2. 栈和队列的逻辑结构定义、抽象数据类型定义和各种存储结构的描述。
3. 栈和队列在两种存储结构上基本操作的实现。
4. 递归算法的设计和运行原理以及递归算法到非递归算法的机械转换。
5. 栈和队列在程序设计中的应用。

3.3.2 教学目标

1. 了解栈和队列的逻辑特点。
2. 熟悉各名词术语的含义，掌握基本概念。
3. 理解并熟练掌握栈和队列的两类存储结构的 C 语言描述。
4. 熟练掌握栈和队列在两种存储结构上基本操作的算法实现。
5. 理解递归算法执行过程中栈的状态变化过程。
6. 理解递归算法到非递归算法的机械转化过程。

7. 能够从时间和空间复杂度的角度综合比较栈和队列两种存储结构的不同特点及其适用场合。
8. 掌握栈和队列在程序设计中的常见应用。

3.3.3 教学重点

1. 栈和队列的两类存储结构的 C 语言描述。
2. 栈和队列在两类存储结构上基本操作的算法实现。
3. 不同存储结构的特点、优缺点和适用的场合。
4. 递归算法的执行原理。
5. 深刻体会递归算法的两个组成部分。
6. 递归算法到非递归算法的机械转化。
7. 栈在程序设计中的常见应用。

3.3.4 教学难点

1. 递归算法到非递归算法的机械转化。有一类递归算法可以用迭代算法转换成非递归算法，但是并不是所有递归算法都可以这样转化成非递归算法。递归算法到非递归算法的机械转化需要深入理解递归工作栈的工作状态。

3.4 串

3.4.1 主要内容

1. 串的结构特性。
2. 串的逻辑结构定义、抽象数据类型定义和各种存储结构的描述。
3. 串的三种存储表示：定长顺序存储结构、块链存储结构和堆分配存储结构。
4. 在三种存储结构上串操作的实现。
5. C 语言中字符串的库函数介绍。
6. 串的模式匹配算法。

3.4.2 教学目标

1. 了解串区别与线性表的特点。
2. 熟悉各名词术语的含义，掌握基本概念。
3. 理解串的三类存储结构的 C 语言描述。
4. 熟悉串的七种基本操作的算法实现，并能利用这些基本操作实现串的其他各种操作的方法。
5. 熟练掌握在串的定长存储结构和堆存储结构上实现各种操作的算法。
6. 理解串匹配的 KMP 算法，熟悉 next 函数的定义，学会手工计算给定模式串的 next 函数值和改进的 next 函数值。
7. 能够从时间和空间复杂度的角度综合比较串三种存储结构的不同特点及其适用场合。
8. 了解串操作的应用方法和特点。

3.4.3 教学重点

1. 与串相关的基本概念。
2. 串三类存储结构的 C 语言描述。
3. 串的一般模式匹配算法和 KMP 模式匹配算法。

3.4.4 教学难点

1. 串匹配的 KMP 算法。KMP 算法原理比较复杂，学生接受比较困难。

3.5 数组和广义表

3.5.1 主要内容

1. 数组的逻辑结构定义、抽象数据类型定义和各种存储结构的描述。
2. 特殊矩阵和稀疏矩阵的压缩存储方法及运算的实现。
3. 广义表的逻辑结构和存储结构、m 元多项式的广义表表示以及广义表的操作递归算法。

3.5.2 教学目标

1. 了解数组和广义表区别与线性表的特点。
2. 熟悉各名词术语的含义，掌握基本概念。
3. 理解并熟练掌握数组顺序存储结构的 C 语言描述。
4. 掌握数组的顺序存储表示和实现。
5. 掌握数组在以行为主的存储结构中的地址计算方法。
6. 掌握对特殊矩阵进行压缩存储时的下标变换公式。
7. 了解稀疏矩阵的两种压缩存储方法的特点和使用范围，领会以三元组表示稀疏矩阵时进行矩阵运算采用的处理方法。
8. 掌握广义表的结构特点及其存储表示方法，掌握对非空广义表进行分解的两种分析方法：即可将一个非空广义表分解为表头和表尾两部分或者分解为 n 个子表。
9. 掌握利用分治法的算法设计思想编制递归算法的方法。

3.5.3 教学重点

1. 数组顺序存储结构的 C 语言描述。
2. 数组在顺序存储结构上基本操作的算法实现。
3. 多维数组的映像函数。
4. 稀疏矩阵存储结构的 C 语言描述。
5. 稀疏矩阵在不同存储结构上基本操作的算法实现。
6. 稀疏矩阵不同存储结构的特点、优缺点和适用场合。
7. 广义表链式储存结构的 C 语言描述。
8. 广义表在链式存储结构上基本操作的算法实现。

3.5.4 教学难点

1. 数组在顺序存储结构上基本操作的算法实现。抽象数组类型的维数是不确定的，因此根据下标访问数组元素的操作需要传递的下标个数也不固定，因此需要用到可变参数函数的编程技术。另外，多维数组中元素存储位置的（映像函数）计算比较抽象。
2. 广义表链式储存结构的 C 语言描述。广义表的结点有两类，因此 C 语言描述时用到

C 语言联合体语法知识，导致广义表链式存储结构的 C 语言描述比较复杂。

3.6 树和二叉树

3.6.1 主要内容

1. 树的定义和基本术语。
2. 二叉树的基本术语和基本性质。
3. 二叉树的逻辑结构定义、抽象数据类型定义和各种存储结构的描述。
4. 二叉树的遍历和线索化以及遍历算法的各种描述形式。
5. 树和森林的定义和各种存储结构的描述。
6. 树和森林与二叉树的转换。
7. 树和森林的遍历。
8. Huffman 树及其应用。

3.6.2 教学目标

1. 熟悉各名词术语的含义，掌握基本概念。
2. 熟练掌握二叉树的结构特性，了解相关性质的证明方法。
3. 理解并熟练掌握二叉树两类存储结构的 C 语言描述。
4. 熟练掌握二叉树在两种存储结构上基本操作的实现算法。
5. 熟悉二叉树的各种存储结构的特点及适用范围。
6. 熟练掌握二叉树的遍历算法，包括递归算法和非递归算法。以及先序、中序、后序和层次遍历算法。
7. 掌握二叉树的线索化算法。
8. 熟悉树的各种存储结构及其特点，掌握树和森林与二叉树的转换方法。
9. 树与森林的遍历方法。
10. 了解最优二叉树的特性，掌握建立最优二叉树和哈夫曼编码的方法。

3.6.3 教学重点

1. 二叉树的性质。
2. 二叉树两类存储结构的 C 语言描述。

3. 二叉树在两类存储结构上基本操作的算法实现。
4. 二叉树的遍历。实现二叉树遍历的具体算法与所采用的存储结构有关。不仅要熟练掌握各种遍历策略的递归和非递归算法，了解遍历过程中“栈”的作用和状态，而且能灵活运用遍历算法实现二叉树的其他操作。层次遍历是按另一种搜索策略进行的遍历。
5. 二叉树的线索化。线索化的实质是建立结点与其相应序列中的前驱或后继的方法。二叉树的线索化过程是基于对二叉树进行遍历，而线索化二叉树又为相应的遍历提供了方便。
6. Huffman 树的构造以及 Huffman 码的生成。

3.6.4 教学难点

1. 二叉树的遍历。实现二叉树遍历的具体算法与所采用的存储结构有关。不仅要熟练掌握各种遍历策略的递归和非递归算法，了解遍历过程中“栈”的作用和状态，而且能灵活运用遍历算法实现二叉树的其他操作。层次遍历是按另一种搜索策略进行的遍历。
2. 二叉树的线索化。

3.7 图

3.7.1 主要内容

1. 图的定义和术语。
2. 图的三种存储结构：数组表示法、邻接表、十字链表/邻接多重表。
3. 图的两种遍历策略：深度优先搜索和广度优先搜索。
4. 图的连通性。
5. 连通分量和最小生成树。
6. 拓扑排序和关键路径。
7. 两类求最短路径问题的解法。

3.7.2 教学目标

1. 熟悉各名词术语的含义，掌握基本概念。

2. 理解并熟练掌握图的各种存储结构的 C 语言描述。
3. 熟练掌握图在各种存储结构上基本操作的算法实现。
4. 熟练掌握图的两种遍历方法：深度优先搜索（递归和非递归）和广度优先搜索。
5. 应用图的遍历算法求解各种简单路径问题。
6. 掌握图的两种最小生成树算法：普里姆（Prim）算法和克鲁斯卡尔（Kruskal）算法。
7. 掌握拓扑排序算法。
8. 掌握图的关键路径算法。
9. 掌握图的最短路径算法：迪杰斯特拉（Dijkstra）算法和弗洛伊德（Floyd）算法。

3.7.3 教学重点

1. 图的三类存储结构的 C 语言描述。
2. 图在三类存储结构上基本操作的算法实现。
3. 图的两种搜索路径的遍历。
4. 图的最小生成树算法。
5. 拓扑排序算法。
6. 图的关键路径算法。
7. 图的最短路径算法。

3.7.4 教学难点

1. 图的三类存储结构的 C 语言描述。
2. 图在三类存储结构上基本操作的算法实现。
3. 图的两种搜索路径的遍历。
4. 图的最小生成树算法。
5. 图的关键路径算法。
6. 图的最短路径算法。

3.8 查找

3.8.1 主要内容

1. 静态查找表的各种实现。

2. 动态查找表的各种实现。
3. 哈希表的各种实现。
4. 各种查找表的查找性能（平均查找长度）分析。

3.8.2 教学目标

1. 熟练掌握普通顺序表和有序表的查找方法。
2. 熟悉静态查找树的构造方法和查找算法，理解静态查找树和折半查找的关系。
3. 熟练掌握二叉排序树的构造和查找方法。
4. 掌握二叉平衡树的维护平衡方法。
5. 理解 B-树的特点及其基本操作、了解 B+树的概念。
6. 熟练掌握哈稀表的构造方法，深刻理解哈稀表与其它结构的表的实质性差别。
7. 掌握描述查找过程的判定树的构造方法，以及按定义计算各种查找方法在等概率情况下查找成功时的平均查找长度。

3.8.3 教学重点

1. 静态查找。
2. 动态查找。
3. 哈希表。

3.8.4 教学难点

1. 平衡二叉树的构造。
2. B-树的特点及其基本操作。
3. 各种查找表的查找性能分析。

3.9 内部排序

3.9.1 主要内容

1. 插入排序的基本思想、算法特点、排序过程及时间复杂度分析。
2. 交换排序的基本思想、算法特点、排序过程及时间复杂度分析。

3. 选择排序的基本思想、算法特点、排序过程及时间复杂度分析。
4. 归并排序的基本思想、算法特点、排序过程及时间复杂度分析。
5. 基数排序的基本思想、算法特点、排序过程及时间复杂度分析。

3.9.2 教学目标

1. 熟悉各名词术语的含义，掌握基本概念。
2. 深刻理解排序的定义和各种排序方法的特点，并加以灵活应用。
3. 了解各种方法的排序过程及其依据的原则。
4. 掌握各种排序方法的时间复杂度的分析方法。能从“关键字间的比较次数”分析排序算法的平均情况和最坏情况的时间性能。

3.9.3 教学重点

1. 五类排序算法的特点。
2. 五类排序算法的基本思想。
3. 五类排序算法的时间复杂度分析。
4. 五类排序算法的优劣比较。
5. 五类排序算法中尤其重点掌握希尔排序、快速排序、堆排序、归并排序及基数排序这些高效排序算法。

3.9.4 教学难点

1. 真正理解各种排序算法的过程。
2. 分析各种排序方法的性能。

第四部分　相关资料

教材

严蔚敏，吴伟民编著．数据结构（C 语言版）．北京：清华大学出版社，2014

参考书目

[1] 严蔚敏，吴伟民，米宁编著．数据结构题集（C 语言版）．北京：清华大学出版社，2006

[2] William Ford, William Topp 著；刘卫东，沈官林译．数据结构 C++语言描述．清华大学出版社，2000

[3] Larry Nyhoff 著；黄明达等译．数据结构与算法分析——C++语言描述（第 2 版）．北京：清华大学出版社，2006

[4] Horowitz, E 等著；李建中，张岩，李治军译．数据结构（C 语言版）．北京：机械工业出版社，2006

[5] Mark Allen Weiss 著；张丽萍译．数据结构与问题求解（C++版）（第 2 版）．北京：清华大学出版社，2005

执笔人：陈润资
审定人：赵书良
批准人：赵书良