**《数据结构与算法》课程教学大纲**

1. **课程名称：** 数据结构与算法
2. **课程代码:** CST21115
3. **课程英文名称：**Data Structures and Algorithms
4. **课程负责人：** 李佳
5. **学时与学分**： 4.5学分，总学时80：理论课64学时
6. **课程性质：** 专业基础课
7. **课程类型:** 必修课
8. **适用专业**： 计算机科学与技术/信息安全/物联网工程
9. **选课对象**： 本科生
10. **预修课程**： 程序设计基础、离散数学、线性代数
11. **使用教材：**
12. **参考书目：**

[1] Clifford A. Shaffer，数据结构与算法分析（C++版）（第三版），电子工业出版社，2013年10月第1版。

[2] T.H. Cormen, et al. Introduction to Algorithms, 3rd Edition, MIT Press, 2015.

1. **开课单位**： 计算机学院
2. **课程描述**

《数据结构与算法》是面向计算机专业学生的专业基础课程。本课程的教学目标是使学生学会在非数值计算数学模型下分析计算机加工数据的理论和方法，以便为应用所涉及的数据选择适当的逻辑结构、存储结构及相应的算法，并初步掌握对算法的时间分析和空间效率分析。课程的教学内容还包括经典的算法设计方法，使学生通过学习能熟练应用课程介绍的算法设计方法来解决软件开发中的实际问题。除了理论教学，课程注重学生编程能力与解决实际问题能力的培养，通过使用在线程序测评系统等实践方式，让学生线上线下挑战各种类型、不同难度的程序设计实践课题。同时，算法设计需要对问题进行分析，对问题的相关方面建模并用最有效的办法实现问题求解，对培养学生的计算思维，即运用计算机科学的基础概念去求解问题、设计系统和理解人类的行为有积极的意义。

1. **课程支撑的毕业要求指标点及课程目标**

**1.本课程支撑的毕业要求指标点**

（1）指标点R1.3：能够运用计算机专业基础知识进行实际工程问题中的计算机算法的设计与实现

（2）指标点R2.3：能认识到解决问题有多种方案可选择，能通过文献研究寻求可替代的解决方案；

（3）指标点R3.2：能够针对特定需求，完成算法和软硬件功能模块的设计；

**2.本课程拟达到的教学目标**

本课程的目标是让学生掌握数据在计算机中的表达、存储以及处理的基本原理和方法，理解计算机算法效率分析与设计所涉及的基本概念和基础知识，掌握基本的算法分析方法和常见的算法设计方法，能熟练应用课程介绍的数据结构与算法解决软件开发中的实际问题；通过分析并解决复杂问题实例，进一步加深学生对复杂数据结构原理、算法设计方法的认识和理解。具体教学目标如下：

1. 学生能够理解和应用线性表、栈、队列、二叉树、图等数据结构的理论和方法；
2. 学生能够理解经典算法原理和算法设计基本概念；
3. 学生能够理解算法分析基础理论；
4. 学生能够应用数据结构和经典的算法设计与分析方法；
5. 学生能够在线上线下解决不同难度的程序设计实践课题。

**3.课程教学目标与毕业要求指标点的关系**

|  |  |  |  |
| --- | --- | --- | --- |
| 课程教学目标 | 毕业要求二级指标点 | | |
| R1.3 | R2.3 | R3.2 |
| 子目标1 | ✓ |  |  |
| 子目标2 | ✓ |  |  |
| 子目标3 |  | ✓ |  |
| 子母标4 |  | ✓ |  |
| 子目标5 |  |  | ✓ |

课程教学子目标1是培养学生能够理解和应用基本数据结构的理论和方法，具体包括：线性表、栈、队列、二叉树、图等数据结构；子目标2是培养学生能够理解分治策略、动态规划、贪心法以及最大流等经典算法的原理及设计过程，因此可以对毕业要求指标点R1.3的达成提供支撑。

课程教学子目标3是培养学生能够理解算法渐进时间及空间复杂度分析的基本理论，理解时间复杂度与空间复杂度之间的权衡原则，根据时间或空间的实际需求选择不同的数据结构和算法解决问题。课程教学子目标4是培养学生能够分析常见的计算问题，通过自主思索并查阅资料发现解决问题所需的数据结构和算法，然后编写程序解决问题并对程序的时间复杂度和空间复杂度进行分析，因此可以对毕业要求指标点R2.3的达成提供支撑。

课程教学子目标5是培养学生能够熟练使用在线程序测评系统，主动挑战系统上不同难度的算法课题，选择合适的数据结构并设计解决课题所需的算法，根据课题要求的计算时间和内存容量分析算法的时间与空间复杂度是否满足条件，对较难的课题积极在网上查找解题思路，最终编写程序并在线提交，因此可以对毕业要求指标点R3.2的达成提供支撑。

1. **课程教学方式/方法**

对于课程教学目标1、目标2和目标3，主要采用课堂教学的方式，通过知识教授、问题讨论等途径，使学生能够理解和应用数据结构理论与方法以及算法分析与设计的基本方法。

对于课程教学目标4，主要采用作业和使用在线程序测评系统进行编程的方式，通过解答思考题并编程解决不同难度的算法课题，训练学生应用合理的数据结构并设计高效的算法，通过编程解决计算问题的能力。

对于课程教学目标5，主要采用上机实验的方式，通过若干编程任务的训练，使学生能够阅读和编写C++程序代码，并能够熟练使用程序开发工具，深入理解数据结构的性质及算法性能。同时，通过撰写实验报告训练学生能够准确、清晰地描述所开发程序的代码结构和功能。

课程采用双语教学的方式。授课老师可选择采用翻转课堂的方式。

1. **课程教学内容与学时**

**1.课堂教学（64学时）**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 一级知识点 | 二级知识点 | 三级知识点 | 掌握层次 | 学时数分配 |
| 1. 课程介绍与基本概念 | 1.1数据结构基本概念 | 无 | 理解 | 2 |
| 1.2算法设计基本概念 | 无 | 理解 |
| 1.3算法分析基本概念 | 无 | 理解 |
| 2. 基本排序与算法分析 | 2.1简单排序法 | 2.1.1排序的基本概念 | 理解 | 1 |
| 2.1.2 插入排序 |
| 2.1.3 冒泡排序 |
| 2.1.4 选择排序 |
| 2.1算法分析基础 | 2.1.1 算法的正确性 | 记忆 | 1 |
| 2.1.2 时间复杂度 | 理解 |
| 2.1.3 空间复杂度 | 理解 |
| 2.1.4 算法的简单性 | 记忆 |
| 2.1.5 算法的最优性 | 记忆 |
| 2.2渐近分析 | 2.2.1 最佳、最差和平均 | 理解 | 2 |
| 2.2.2 Big Theta | 理解 |
| 2.2.3 Big O | 理解 |
| 2.2.4 Big Omega | 理解 |
| 2.2.5程序的渐进复杂度分析方法 | 应用 |
| 2.3 归并排序 | 2.3.1 分治法思想 | 记忆 | 1 |
| 2.3.2 归并排序 | 应用 |
| 2.3.3归并排序的时间复杂度分析 | 理解 | 2 |
| 2.3.4 递推方程求解：展开法、代入法与递推树 | 理解 |
| 2.4 快速排序 | 2.4.1 两种划分 | 应用 | 2 |
| 2.4.2 平均时间复杂度 | 理解 |
| 2.4.3 三数取中划分 | 应用 |
| 4.2.3 改进算法的时间复杂度分析 | 理解 |
| 3.线性结构 | 3.1线性表的顺序表示与实现 | 3.1.1顺序表的定义 | 理解 | 1 |
| 3.1.2顺序表上基本操作的实现 | 应用 |
| 3.2线性表的链式表示和实现 | 3.2.1单链表的定义 | 理解 | 1 |
| 3.2.2单链表上基本操作的实现 | 应用 |
| 3.2.3 空闲链表 | 理解 |
| 3.3 循环链表和双向链表 | 3.3.1 循环链表 | 理解 | 1 |
| 3.3.2 双向链表 | 理解 |
| 3.4 栈 | 3.4.1 栈的逻辑定义 | 理解 | 2 |
| 3.4.2 顺序栈 | 理解 |
| 3.4.3 链栈 | 了解 |
| 3.4.4 栈的应用： 数制转换、括号匹配、表达式转换等 | 应用 | 2 |
| 3.5 队列 | 3.5.1 队列的逻辑定义 | 理解 | 2 |
| 3.5.2 循环队列 | 应用 |
| 3.5.3 链式队列 | 了解 |
| 3.6 数组与压缩存储 | 3.6.1数组的逻辑定义 | 理解 | 2 |
| 3.6.2特殊矩阵的压缩存储 | 记忆 |
| 3.6.3 稀疏矩阵的压缩存储 | 记忆 |
| 3.6.4 快速置换 | 理解 |
| 4.树与二叉树 | 4.1 树的定义和基本术语 | 无 | 记忆 | 2 |
| 4.2 二叉树 | 5.2.1 二叉树的定义与基本性质 | 理解 |
| 5.2.2二叉树的存储结构 | 理解 |
| 4.3 二叉树遍历 | 4.3.1 遍历的分类 | 理解 | 2 |
| 4.3.2 递归算法 | 应用 |
| 4.3.3 非递归算法（栈的应用） | 应用 |
| 4.4 树和森林 | 4.4.1 树的存储结构 | 记忆 | 1 |
| 4.4.2 树和森林的遍历 | 记忆 |
| 4.4.3 森林与二叉树的转换 | 理解 |
| 5. 查找 | 5.1静态查找 | 5.1.1 查找基本概念 | 理解 | 2 |
| 5.1.2 顺序查找 | 应用 |
| 5.1.3 二分查找 | 应用 |
| 5.2 字典 | 5.2.1 字典的定义 | 应用 | 2 |
| 5.2.2 字典的实现 | 理解 |
| 5.3 哈希表 | 5.3.1 哈希表基本原理 | 记忆 | 2 |
| 5.3.2 哈希函数的构造方法 | 记忆 |
| 5.3.3 处理冲突的方法 | 理解 |
| 5.4 动态查找 | 5.4.1二叉查找树 | 应用 | 2 |
| 5.4.2 自平衡二叉查找树 | 理解 |
| 6. 先进排序 | 6.1 希尔排序 | 6.1.1 希尔排序 | 理解 | 1 |
| 6.1.2 时间复杂度分析 | 记忆 |
| 6.2 线性时间排序 | 6.2.1 计数排序 | 理解 | 2 |
| 6.2.2 基位排序 | 理解 |
| 6.2.3 桶排序 | 记忆 |
| 6.3 堆排序 | 6.3.1 堆的逻辑定义 | 理解 | 2 |
| 6.3.2 堆的实现方法 | 应用 |
| 6.3.3 堆排序及时间复杂度分析 | 理解 |
| 7.动态规划 | 7.1 动态规划基本思想 | 7.1.1 最优子结构 | 理解 | 1 |
| 7.1.2 子问题重叠 | 记忆 |
| 7.1.3 备忘录与迭代 | 理解 |
| 7.2 动态规划实例 | 7.2.1 Rod Cutting Problem | 应用 | 5 |
| 7.2.2背包问题 | 应用 |
| 7.2.3 矩阵连乘 | 应用 |
| 7.2.4最长公共子序列 | 应用 |
| 7.2.5最优二叉搜索树 | 理解 |
| 7.2.6序列对齐 | 理解 |
| 8.贪心算法 | 8.1 贪心法基本思想 | 8.1.1 最优子结构 | 记忆 | 2 |
| 8.1.2 贪心选择 | 理解 |
| 8.1.3 活动选择问题 | 应用 |
| 8.2哈夫曼编码 | 8.2.1 前缀树的定义 | 记忆 | 2 |
| 8.2.2哈夫曼算法的最优性证明（贪心选择） | 理解 |
| 8.2.3 哈夫曼算法实现（堆的应用） | 应用 |
| 9.图 | 9.1图的定义及术语 | 无 | 理解 | 1 |
| 9.2 图的存储结构 | 9.2.1 邻接矩阵 | 应用 |
| 9.2.2 邻接链表 | 应用 |
| 9.3 图的遍历 | 9.3.1 深度优先 | 应用 | 2 |
| 9.3.2 广度优先（队列的应用） | 应用 |
| 9.3.3 拓扑排序 | 理解 |
| 9.4 单源最短路径 | 9.4.1 最短路径的性质 | 记忆 | 3 |
| 9.4.2 Dijkstra算法的正确性（贪心选择） | 理解 |
| 9.4.3 Dijkstra算法的实现（堆应用） | 应用 |
| 9.4.4 Bellman-Ford算法 | 理解 |
| 9.4.5 时间复杂度分析 | 记忆 |
| 9.5 所有顶点对间最短路径 | 9.5.1 矩阵乘法（动态规划+快速幂） | 理解 | 2 |
| 9.5.2 Floyd-Warshall算法（动态规划） | 应用 |
| 9.5.3 时间复杂度分析 | 记忆 |
| 9.6 最小生成树 | 9.6.1 并查集的逻辑定义与实现 | 应用 | 3 |
| 9.6.2 最小生成树的性质（贪心选择） | 记忆 |
| 9.6.3 Prim算法 | 理解 |
| 9.6.3 Kruskal算法 | 应用 |
| 9.6.4 时间复杂度分析 | 记忆 |
| 10. 流量图 | 10.1 网络流图 | 10.1.1网络流图的基本概念及基本定理 | 记忆 | 2 |
| 10.1.2截集与截量 | 记忆 |
| 10.1.3剩余网络 | 理解 |
| 10.1.4增广路径 | 理解 |
| 10.1.5最大流—最小截集定理 | 记忆 |
| 10.2最大流 | 10.2.1 Ford-Fulkerson算法 | 应用 | 2 |
| 10.2.2 Edmonds-Karp 算法 | 理解 |
| 10.2.3 时间复杂度分析 | 记忆 |

注释1：课程内容按照相对独立性，用层次结构列出1、2、3级知识点单元。

注释2：对知识学习的要求一般分为“记忆，理解，应用”三个层次。

1. **能承担此课的教师**

李佳、邹东升、何中市、夏云霓、秦红星、陈自愈、何静媛

教学大纲制订者：李佳

教学大纲审定者：葛亮