考试题型：

一、简答题（10 分×2 题=20 分）

二、算法题（10 分×6 题=60 分）

三、综合题（20 分）

第一章 算法求解基础

λ 算法的概念

λ 算法特征（输入、输出、确定性、可行性、有穷性）——掌握每种特征的含义、算法和程序的区别

λ 描述算法的方法（自然语言、流程图、伪代码、程序设计语言）

λ 欧几里德算法（辗转相除法）——递归/迭代程序实现及其变形

λ 常见算法种类——精确算法、启发式算法、近似算法、概率算法

λ 数学归纳法证明；

第二章 算法分析基础

λ 算法复杂度——运行一个算法所需的时间和空间。

λ 好算法的四个特征（正确性、简明性、效率、最优性）

正确性 vs 健壮性 vs 可靠性

最优性——算法（最坏情况下）的执行时间已达到求解该类问题所需时间的下界。

λ 影响程序运行时间的因素（程序所依赖的算法、问题规模和输入数据、计算机系统性能）

λ 算法的渐近时间复杂度 ——数量级上估计（O、Ω、Θ）

λ 最好、最坏、平均时间复杂度——定义

——课后习题 2-8（通过考察关键操作的执行次数）

λ 时间复杂度证明

——课后习题 2-10，2-13，2-17

λ 算法按时间复杂度分类：多项式时间算法、指数时间算法

多项式时间算法：**$O(1)<O(\log n)<O(n)<O(n\log n)<O(n^2)<O(n^3)$**

指数时间算法：**$O(2^n)<O(n!)<O(n^n)$**

λ 使用主定理求解递归算法的时间复杂度

第五章 分治法

λ 分治法——求解的基本要素：将一个难以直接求解的复杂问题分解成若干个规模较小、相互独立但类型相同的子问题，然后求解这些子问题；如果这些子问题还比较复杂而不能直接求解，还可以继续细分，直到子问题足够小，能够直接求解为止；最后将子问题的解组合成原始问题的解。这种问题求解策略称为分治法。

λ 分治法很自然的导致一个递归算法。

λ 分治法的算法的控制结构框架，平衡子问题思想

λ 递归算法的时间复杂度分析:

递推式 **$T(n)=aT(n/b)+cn^k$，$T(1)=c$**

——递推式中每部分的含义

——求解得到算法的渐近时间复杂度（分三种情况）

——改进思路

λ 求最大最小元

λ 二分搜索算法框架

λ 对半搜索

——程序实现

——对半搜索二叉判定树（树的构成）

——对半搜索二叉判定树性质（左右子树结点数、树高等）

——对半搜索的时间复杂度分析（搜索成功/失败、最好/最坏/平均）。

◆ 二叉判定树的性质→对半搜索的时间复杂度

成功搜索：平均、最坏 **O(logn)**

失败搜索：平均、最好、最坏都是 **Θ(logn)**

◆ （通过关键字值间的比较，搜索指定关键字值元素）这类搜索算法最坏情况下的时间下界为 **O(logn)**，因此对半搜索是最优算法。

——课后习题 **5-8**

λ 最优算法

λ 两路合并排序

——分治法排序过程

——程序实现

——时间复杂度分析（最好、最坏、平均）

——空间复杂度分析

——是否最优算法

λ 快速排序

——排序过程（每一趟分划后的排列和子问题划分）

——时间复杂度分析（最好、最坏递推式、平均）

——课后习题 **5-11**

λ 斯特拉森矩阵乘法——求解时间的递推关系式，时间复杂度，相

应的证明。

λ 棋盘覆盖问题

——算法设计思想

——时间复杂度分析

——最优算法

λ 实验补充——（线性时间选择算法）寻找第 **k** 个最小元

第六章 贪心法

λ （求解最优化问题）贪心法的基本要素：

■ 最优子结构性质

■ 贪心选择性质

λ 一般背包问题

——课后习题 **6-1**

λ 最佳合并模式

——最小带权外路径长度

——课后习题 **6-8**，**k** 路最优合并，虚结点

λ 最小代价生成树

——**Prim** 和 **Kruskal** 算法（构造过程、区别）

——共同的理论基础：**MST** 性质

——不同点和应用场合

——课后习题 **6-9**

——**Prim** 算法求解过程（邻接表、**lowcost\nearest\mark** 数组、

输出的边集、构造的最小代价生成树）

λ 补充---聚类间隔的定义

第七章 动态规划法

λ （求解最优化问题）动态规划法的基本要素：

- 最优子结构性质
- 重叠子问题性质

λ 动态规划法求解思路——自底向上、保存子问题的解

λ 动态规划法求解步骤：

- 1）刻画最优解的结构特性
- 2）递归定义最优解值
- 3）以自底向上方式计算最优解值
- 4）根据计算得到的信息构造一个最优解

λ 备忘录方法

1）动态规划法的变形。

2）采用分治法的思想。

3）自顶向下直接递归的方式求解。

——两者的异同和适用场合

λ 多段图问题

——从后向前/从前向后递推式

——结点的 cost 和 d 值求解过程

——最短路径长度，并根据 d 值构造最短路径

注意：d[j]的含义和最短路径的构造。

——课后习题 7-1，7-2

λ 关键路径问题

——earliest、latest 的递推式和求解过程

——寻找关键活动

——构造关键路径

最长路径长度——完成工程的最短时间

——程序实现

——补充题

注意：应由关键活动<i,j>（而不是事件 i）来确定关键路径。

λ 弗洛伊德算法

——d_k 数组和 $path_k$ 数组更新的递推式

——每次迭代后的 d 数组和 path 数组元素值

——程序实现和时间复杂度

λ 最长公共子序列问题

——c 和 s 数组元素的求解递推式

——c 和 s 数组元素求值，得最优解值

——回溯构造最优解

——程序实现

——课后习题 7-9

λ 0/1 背包问题动态规划法求解——自底向上，递推式

λ 0/1 背包问题（实数重量）阶跃点集合求解——非启发式方法、启

发式方法

注意：被支配的阶跃点和所有 **X>M** 的阶跃点均应该去除。

——课后习题 **7-15**、补充题

- 实验补充——装载问题。

第八章 回溯法

- 状态空间树——描述问题解空间的树形结构
  - 问题状态（树中每个结点）
  - 解状态（候选解元组）
  - 答案状态（可行解元组）
  - 最优答案结点（目标函数取最优值的答案结点）
- 回溯法和分支限界法都是通过搜索问题的状态空间树求解
- 比较回溯法与分支限界法的异同。
- 剪枝函数（约束函数、限界函数）——可以剪去不必要搜索的子树，压缩问题求解所需要实际生成的状态空间树的结点。
  - 约束函数——剪去不含答案状态（可行解）的子树
  - 限界函数——剪去不含最优答案结点的子树
- 约束函数（显式约束、隐式约束）
  - 显示约束——规定了所有可能的元组，组成问题的候选解集（解空间）
    - 排列树——用于确定 **n** 个元素的排列满足某些性质的状态空间树，一般有 **n!**个叶结点（解状态）。

- 子集树——从 n 个元素的集合中找出满足某些性质的子集的状态空间树，一般有 2^n 个解状态。

■ 隐式约束——给出了判定一个候选解是否为可行解的条件。

λ 回溯法深度优先搜索问题的状态空间树，用剪枝函数（往往是约束函数）进行剪枝，通常求问题的一个或全部可行解

λ 蒙特卡罗(Monte Carlo)算法——估计回溯法处理一个实例时，状态空间树上实际生成的结点数的方法：$m=1+m_0+m_0m_1+m_0m_1m_2+...$

λ n-皇后问题

——算法思想和程序实现

——显式约束、隐式约束条件，显式约束对应的状态空间树

——Monte Carlo 算法估计实际生成的状态空间树结点数

——补充题（蒙特卡罗方法）

λ 子集和数问题（画出状态空间树）

——（可变长度解、固定长度解）对应的状态空间树不同（P183-184）

——约束函数定义

——课后习题 8-2（固定长度解元组）

λ 图的 m-着色问题——四色定理（四种颜色可对地图着色），如何将地图转化为平面图?

-----算法程序思想和实现

λ 哈密顿环——课后习题 8-10

第九章 分枝限界法

λ 分枝限界法广度优先搜索问题的状态空间树，用剪枝函数（往往是限界函数）进行剪枝，通常求问题的最优解

λ 根据活结点表采用的数据结构不同，分枝限界法分类：

- **FIFO** 分枝限界法
- **LIFO** 分枝限界法
- **LC** 分枝限界法

λ 十五谜问题（定理 **9-1**：判定初始状态是否可以到达目标状态）

——**FIFO**、**LIFO**、**LC** 分枝限界法

——**LC** 分枝限界法中搜索代价 $\hat{c}(x)$ =**f(x)**+ $\hat{g}(x)$

——补充题（十五谜问题的 **LC** 分支限界法求解，生成的状态空间树）

λ 上、下界函数——与最优化问题的目标函数有关

- $\hat{c}(x)$ 是代价函数 **c(X)**的下界函数
- $u(x)$ 是代价函数 **c(X)**的上界函数

λ 如何用上、下界函数进行剪枝：

① （若目标函数取最小值时为最优解）

- 则用上界变量 **U** 记录迄今为止已知的最小代价上界（即迄今为止已知的可行解中目标函数最小值），可以确定最小代价答案结点（最优解）的代价值不会超过 **U**。
- 对任意结点，若 $\hat{c}(X)$ ≥U，则 X 子树可以剪枝。
- 为了不至误剪去包含最小代价答案结点的子树，若 **X** 代表部分向量，则 **U=min {u(X)+** ε **,U}**；若 **X** 是答案结点，则 **U=min**

{cost(X),U}。

②（若<u>目标函数取最大值</u>时为最优解）

■ 则用下界变量 **L** 记录迄今为止已知的最大代价下界（即迄今为止已知的可行解中目标函数最大值），最大代价答案结点（最优解）的代价值不会小于 **L**。

■ 对任意结点，若 $u(X)$ ≤L，则 X 子树可以剪枝。

■ 为了不至误剪去包含最大代价答案结点的子树，若 **X** 代表部分向量，则 **L=max {**$\hat{c}(x)-\varepsilon$ **,L}**；若 **X** 是答案结点，则 **L=max {cost(X),L}**。

λ 带时限的作业排序

——前提：作业按时限排序，以便判断是否可行。

——画出 **JSFIFOBB** 算法（**FIFO** 分枝限界法）实际生成的状态空间树（目标函数为损失最小）

——从活结点表中<u>选取扩展结点时，应保证扩展结点满足</u> <u>$\hat{c}<U$，否则剪枝</u>。

——扩展结点<u>生成孩子时，应剪去不可行的孩子结点</u>（即：子集内的作业不能在时限内完成）

——对于可行的孩子结点，<u>进一步计算其损失下界 $\hat{C}$ 和损失上界 **u**</u>。<u>当 $\hat{c}<U$ 时生成该结点，否则剪枝</u>。

——每生成一个孩子，需同时<u>检查是否要用 **u** 更新上界变量值 **U**</u>。

——求最优解值（最大作业收益=所有作业收益之和-最优解对

应的最小损失）和最优解（入选的作业编号，可变长度解）

——课后习题 9-2

λ **LC 分支限界法求解 TSP 问题。**

下界：**cc** 为当前结点费用

上界：**lcost=cc+rcost**，优先级

rcost 为当前顶点最小出边费用加上剩余

所有顶点的最小出边费用和。

Bestc:当前当前最优回路的费用

剪枝函数：**lcost>=bestc**

第十章 NP 完全问题

λ 不确定算法及其时间复杂度

λ 最优化问题与判定问题间的转换关系

λ 理解 P 类问题、NP 类问题、NP 难度问题、NP 完全问题的概念

λ 什么是多项式约化？

——课后习题 10-6、10-7

λ 了解 Cook 定理的内容（Steven Cook，1971 年，证明了可满足性问题是 NP 完全的）

——什么是可满足性问题？

λ NP 难度（NP 完全）问题的证明步骤（例：最大集团问题）

——课后习题 10-8

第十三章 密码算法（5%）

λ 信息安全的目标

- 机密性——加密
- 完整性——消息摘要
- 抗否认性——数字签名
- 可用性

λ 现代密码学的两个分支（密码编码学、密码分析学）

λ 两种密码体制（对称密码体制、非对称密码体制）的加/解密原理及优缺点

λ RSA 算法的理论基础、加/解密原理、用途和安全性