**第1课：算法基础与简单排序入门**

* **目标：** 建立算法效率概念，掌握最基础算法。
* **内容：**
  + 什么是排序？稳定性概念 | 时间复杂度O(n²)引入
  + **选择排序 (Selection Sort):**
    - **可视化演示** + **手动模拟** (找最小/交换)
    - 核心代码讲解 (**双重循环，找min\_index**)
    - 稳定性分析
  + **冒泡排序 (Bubble Sort):**
    - **可视化演示** (气泡上浮) + **手动模拟**
    - 核心代码讲解 (**相邻比较交换，标志位优化**)
    - 与选择排序对比 (交换次数)
* **练习：** 手动模拟对[5, 3, 8, 1]进行选择排序和冒泡排序。写出核心循环代码框架。

**第2课：插入的力量与高级入门**

* **目标：** 理解插入思想，了解希尔优化雏形。
* **内容：**
  + **插入排序 (Insertion Sort):**
    - **可视化演示** (理扑克牌) + **手动模拟**
    - 核心代码讲解 (**从后向前比较插入**)
    - 分析：为何对**部分有序**数据高效？(接近O(n))
    - 与选择/冒泡对比
  + **希尔排序 (Shell Sort) - 概念与思想：**
    - 引入：如何改进插入排序？
    - **思想：** 分组插入 (递减增量序列，如**gap = n/2, n/4, ... 1**)
    - **可视化演示** (感受分组排序如何让元素大步移动)
    - **重点讲思想，不深究代码/复杂序列**
* **练习：** 对[7, 6, 5, 4, 3, 2, 1]手动模拟插入排序。简述希尔排序的分组思想。

**第3课：分治法的威力 - 归并排序 (上)**

* **目标：** 掌握分治思想，理解归并流程。
* **内容：**
  + **分治法 (Divide and Conquer) 精讲：** 分解 -> 解决 -> 合并
  + **归并排序 (Merge Sort) - 原理与合并：**
    - **可视化演示** (分割与合并动画)
    - **核心：合并两个有序数组 (merge函数)**
      * 手动模拟合并 [1, 3, 5] 和 [2, 4, 6]
      * 代码精讲 (**双指针，新数组暂存**)
* **练习：** 实现**merge**函数。手动模拟合并过程。

**第4课：分治法的威力 - 归并排序 (下) 与效率飞跃**

* **目标：** 实现完整归并排序，理解O(n log n)意义。
* **内容：**
  + **归并排序完整实现：**
    - 递归分解 (**merge\_sort**函数)
    - 代码精讲 (递归终止条件、递归调用、调用**merge**)
    - 空间复杂度分析 (O(n))
  + **算法效率革命：O(n log n) vs O(n²)**
    - 画递归树理解层数(log₂n)与每层工作量(n)
    - **实验演示：** 对比归并 vs 插入/选择 在10000个随机数上的速度 (代码计时)
* **练习：** 补全递归框架代码。解释为什么归并排序是O(n log n)。

**第5课：排序王者 - 快速排序 (上) 核心思想**

* **目标：** 理解快排核心：分区(Partition)。
* **内容：**
  + **快速排序 (Quick Sort) 思想：** 分治 + 分区
  + **核心：分区操作 (partition函数) - Lomuto/Hoare选一精讲(推荐Hoare)**
    - **可视化演示** (基准值(pivot)选择，双指针扫描交换)
    - **手动模拟** 分区过程 (e.g., [6, 2, 7, 3, 9, 1], pivot=6)
    - 代码精讲 (指针移动、交换条件、最终pivot位置确定)
    - 稳定性分析
* **练习：** 手动模拟Hoare分区过程。写出**partition**函数核心交换逻辑。

**第6课：排序王者 - 快速排序 (下) 实现与优化**

* **目标：** 实现完整快排，了解关键优化。
* **内容：**
  + **快速排序完整实现：**
    - 递归框架 (**quick\_sort**：调用**partition**，递归左右区间)
    - 代码精讲
  + **关键优化：**
    - **随机化pivot：** 为何能避免最坏O(n²)? (代码实现：**swap(arr[low], arr[random\_index])**)
    - (可选) 小区间优化 (区间<15用插入排序)
  + **时间复杂度分析：** 平均O(n log n)，最坏O(n²)及随机化避免
* **练习：** 实现随机化pivot选择。简述随机化的重要性。

**第7课：基于二叉堆的排序 - 堆排序**

* **目标：** 理解堆结构，掌握堆排序流程。
* **内容：**
  + **二叉堆 (Binary Heap) 精要：**
    - 完全二叉树结构
    - 大顶堆/小顶堆定义
    - **heapify**操作 (下沉sift down) - **可视化演示维护堆性质**
  + **堆排序 (Heap Sort) 步骤：**
    1. 建堆 (Heapify: 从最后一个非叶节点向上调整) - **手动模拟**
    2. 交换堆顶与末尾元素
    3. 堆大小减1，对堆顶元素**sift down**
    4. 重复2-3
    - **可视化演示全过程**
    - 核心代码讲解 (**heapify**, **build\_heap**, 排序循环)
  + 时间复杂度O(n log n)，空间O(1)，不稳定
* **练习：** 手动模拟对[4, 10, 3, 5, 1]建大顶堆过程。写出一次**sift\_down**操作。

**第8课：突破比较的界限 - 线性排序 (上) 计数与桶**

* **目标：** 理解非比较排序思想及应用限制。
* **内容：**
  + **引入：** 比较排序下限O(n log n)，非比较排序如何突破？
  + **计数排序 (Counting Sort)：**
    - **适用场景：** 整数、范围小(**k=O(n)**）
    - **原理：** 统计频率 -> 计算前缀和 -> 确定位置
    - **可视化演示** + **手动模拟** (e.g., [2, 5, 3, 0, 2, 3, 0, 3])
    - 核心代码讲解
    - 稳定性实现要点
  + **桶排序 (Bucket Sort)：**
    - **适用场景：** 均匀分布的数据
    - **原理：** 分桶 -> 桶内排序 (插入等) -> 合并
    - **可视化演示** (分桶与合并)
    - 强调是**框架思想**，桶内排序方法可自由选择
* **练习：** 对[1, 4, 1, 2, 7, 5, 2]手动模拟计数排序。简述桶排序步骤。

**第9课：突破比较的界限 - 线性排序 (下) 基数排序**

* **目标：** 掌握多关键字排序思想。
* **内容：**
  + **基数排序 (Radix Sort)：**
    - **适用场景：** 整数、字符串 (等长或补齐)
    - **原理：** LSD (次位优先) / MSD (主位优先) - **重点讲LSD**
    - **核心：** 按位排序 (必须稳定！通常用计数排序作为子过程)
    - **可视化演示** LSD过程 (e.g., [329, 457, 657, 839, 436, 720, 355])
    - 代码框架讲解 (循环位数，调用稳定子排序)
    - 时间复杂度O(d\*(n+k))，d为位数
  + **对比总结桶/计数/基数：** 思想、适用场景、限制 (数据特征要求高)
* **练习：** 对[170, 45, 75, 90, 802, 24, 2, 66]手动模拟LSD基数排序。为什么子排序必须稳定？

**第10课：终极对决 - 算法总结、对比与实战**

* **目标：** 全局梳理，综合应用。
* **内容：**
  1. **算法全家福回顾：** (快速过一遍所有算法核心思想和关键点，用对比表格强化)
  2. **终极对比表深度解析：**
     + 时间复杂度 (最好/平均/最坏)
     + 空间复杂度
     + 稳定性
     + 适用场景/数据特点
     + In-place? (堆排、快排、希尔、冒泡、选择、插入是；归并、计数、桶、基数一般不是)
  3. **实战挑战：**
     + **场景选择题：** 给定数据特点 (大量重复整数？近乎有序？范围已知？内存紧张？要求稳定？)，选择最优算法并解释。
     + **小数据集手动排序竞赛：** 抽签用指定算法对同一组数据手动排序，比速度和准确性。
     + (可选) **代码填空大赛：** 补全关键算法缺失的几行核心代码。
  4. **学习总结与展望：** 强调算法思想的重要性，推荐进阶资源 (如《算法导论》、LeetCode)。

时间：课后作业讲解(5min) + 可视化(5min) + 原理讲解(5min) + 代码精讲(15min) + 练习(20min) + 答疑(10min) = 50min。

每节课结束后下发**注释详尽的代码框架，课程后留提升作业，手写代码，便于检查学生思维逻辑能力与错误规避能力。**

课堂软件：Jupyter Notebook

安装方法：首先安装python

打开cmd，分别输入：

pip install notebook

jupyter notebook