1. 可视化展示冒泡排序的过程:

- 设计一个交互式的界面,允许用户输入一个未排序的数组。

- 使用动画或者图形化的方式,逐步展示冒泡排序的每一轮排序过程。

- 在每一步交换元素时,使用不同的颜色或者特效来突出显示当前正在比较的元素。

2. 提供步骤控制和说明:

- 在可视化界面上,提供 "下一步" 和 "上一步" 的按钮,让用户能够控制排序过程的进度。

- 在每一步骤执行时,提供文字说明,解释当前步骤的操作,例如 "比较元素 5 和 3,由于 5 > 3,交换它们的位置"。

- 允许用户暂停、继续或者重新开始排序过程。

3. 提供代码展示和解释:

- 在可视化界面的同时,展示冒泡排序的代码实现。

- 当用户单击某一行代码时,在可视化界面上高亮显示当前正在执行的步骤。

- 对关键的代码行提供注释或者解释,帮助用户理解算法的实现逻辑。

4. 支持用户交互和练习:

- 提供一个 "自己尝试" 的模式,让用户可以手动尝试对数组进行冒泡排序。

- 在用户操作的过程中,提供实时的反馈和提示,指导用户完成排序。

- 对用户完成的排序结果进行评估,给出正确性和效率的反馈。

5. 提供多个练习案例和难度选择:

- 准备一系列不同长度和难度的数组作为练习案例。

- 让用户可以选择不同的难度等级,从简单的数组开始练习,逐步过渡到更复杂的案例。

- 记录用户的练习进度和表现,提供学习进度追踪和成就系统。

6. 整合学习资源和评估:

- 提供冒泡排序的算法介绍、时间复杂度分析等学习资料,方便用户系统地学习和理解算法。

- 设计交互式的小测验或者挑战题,评估用户对冒泡排序的掌握程度。

- 根据用户的表现,提供个性化的学习建议和推荐练习。

用户故事：

**作为一名算法初学者,我希望平台提供生动形象的算法可视化演示,让我能直观地理解冒泡排序的运行过程,加深对算法概念和原理的掌握。**

关键验收测试:

选择平台上的一个基础算法,如冒泡排序。

查看该算法的可视化演示,验证演示是否生动形象,能否清晰展示算法的关键步骤和数据变化过程。

在演示过程中,验证是否提供了算法每一步的文字说明,帮助理解算法原理。

改变输入数据(如数组长度、元素大小),验证可视化演示是否能够动态适应不同的数据规模。

原始数组: [1, 6, 5, 3]

第一轮排序:

步骤 1: [1, 6, 5, 3] → [1, 6, 5, 3]

步骤 2: [1, 6, 5, 3] → [1, 5, 6, 3]

步骤 3: [1, 5, 6, 3] → [1, 5, 3, 6]

第一轮排序后: [1, 5, 3, 6]

第二轮排序:

步骤 1: [1, 5, 3, 6] → [1, 5, 3, 6]

步骤 2: [1, 5, 3, 6] → [1, 3, 5, 6]

第二轮排序后: [1, 3, 5, 6]

第三轮排序:

步骤 1: [1, 3, 5, 6] → [1, 3, 5, 6]

第三轮排序后: [1, 3, 5, 6]

冒泡排序完成,最终结果: [1, 3, 5, 6]

冒泡 插入 快速

Id Insert change 之前位置 之后位置 curr\_list 题目序号 步骤数 出题人

FindSortbyId

Nextstep

User全系列

FindCurrList

FindSolution

AddSort(List, type)

deleteSortbyId(id)

modifySort(id, newList)

Entity

Service Controller + test

**作为一名教师,我希望平台提供丰富的算法案例库,包含经典算法和常见算法题,方便我在课堂上进行演示教学。同时,我希望平台支持自定义数据,以便我针对教学需求设计案例。**

关键验收测试:

查看平台的算法案例库,验证是否覆盖了主要的排序、搜索、图论、动态规划等经典算法类型。

查看平台的算法题库,验证是否包含了面试中常见的算法题,如字符串操作、数组查找等。

选择一个算法案例,验证案例的描述是否清晰,可视化演示是否易于理解,是否适合用于课堂教学。

尝试在平台上创建自定义教学案例:

输入自定义的数据,如一个待排序的数组。

验证平台是否支持对自定义数据进行可视化演示。

保存自定义案例,并尝试再次打开,验证案例是否能被正确存储和加载。

邀请一名学生试用自定义案例,获取反馈,评估自定义案例是否达到预期的教学效果。