《算法设计与分析》实验教学大纲

课程编号：CS205102

课程名称：算法设计与分析　　　　英文名称：Design and Analysis of Algorithms

学分/学时：2.5/40　　　　　　　课程性质：专业选修

适用专业：计算机科学与技术　　　建议开设学期：5

物联网工程，卓越，教改

先修课程：离散数学，数据结构　　　开课单位：计算机学院

JAVA 程序设计

一、实验简介

本实验要求学生能够综合运用排序、搜索、图处理和字符串处理的基础算法和数据结构，将算法理论、算法工程和编程实践相结合开发相应的软件，解决科学、工程和应用环境下的实际问题。使学生能充分运用并掌握算法设计与分析的方法以及算法工程技术，为从事计算机工程和软件开发等相关工作打下坚实的基础。

二、实验课程目标与毕业要求

通过本课程的学习使学生系统掌握算法设计与分析的基本概念和基本原理，理解排序、搜索、图处理和字符串处理的算法设计理论及性能分析方法，掌握排序、搜索、图处理和字符串处理的数据结构与算法实现技术。课程强调算法的开发及 Java 实现，理解相应算法的性能特征，评估算法在应用程序中的潜在性能。

三、实验内容及基本要求

（二）几种排序算法的实验性能比较

实现插入排序（Insertion Sort，IS），自顶向下归并排序（Top-down Mergesort，TDM），自底向上归并排序（Bottom-up Mergesort，BUM），随机快速排序（Random Quicksort，RQ），Dijkstra 3-路划分快速排序（Quicksort with Dijkstra 3-way Partition，QD3P）。在你的计算机上针对**不同输入规模数据**进行实验，对比上述排序算法的时间及空间占用性能。要求对于每次输入运行 10 次，记录每次时间/空间占用，取平均值。

Comparison of running time of sorting algorithms (in Micro Seconds)

	Run1	Run2	Run3	Run4	Run5	Run6	Run7	Run8	Run9	Run10	Average
IS											
TDM											
BUM											
RQ											
QD3P											

Comparison of space usage of sorting algorithms (in Kilo Bytes)

	Run1	Run2	Run3	Run4	Run5	Run6	Run7	Run8	Run9	Run10	Average
IS											
TDM											
BUM											
RQ											
QD3P											

回答以下问题：

1. Which sort worked best on data in constant or increasing order (i.e., already sorted data)? Why do you think this sort worked best?

2. Did the same sort do well on the case of mostly sorted data? Why or why not?

3. In general, did the ordering of the incoming data affect the performance of the sorting algorithms? Please answer this question by referencing specific data from your table to support your answer.

4. Which sort did best on the shorter (i.e., *n* = 1,000) data sets? Did the same one do better on the longer (i.e., *n* = 100,000) data sets? Why or why not? Please use specific data from your table to support your answer.

5. In general, which sort did better? Give a hypothesis as to why the difference in performance exists.

6. Are there results in your table that seem to be inconsistent? (e.g., If I get run times for a sort that look like this {1.3, 1.5, 1.6, 7.0, 1.2, 1.6, 1.4, 1.8, 2.0, 1.5] the 7.0 entry is not consistent with the rest). Why do you think this happened?