Project: CS/CE/SE 3345: Data Structures and Algorithm Analysis

Purpose: Analysis of Sorting Techniques

Overview: One of the most important ADTs is the Dictionary and one of the most studied problem is sorting. In this assignment, you will analyze multiple implementations of sorting algorithms.

Write a program to perform analysis on various sorting algorithms utilizing 4 different data types. insertionSort.java, selectionSrot.java, quicksort.java, HeapSort.java, RadixSort.java, and MergeSort.java files are provided with this assignment.

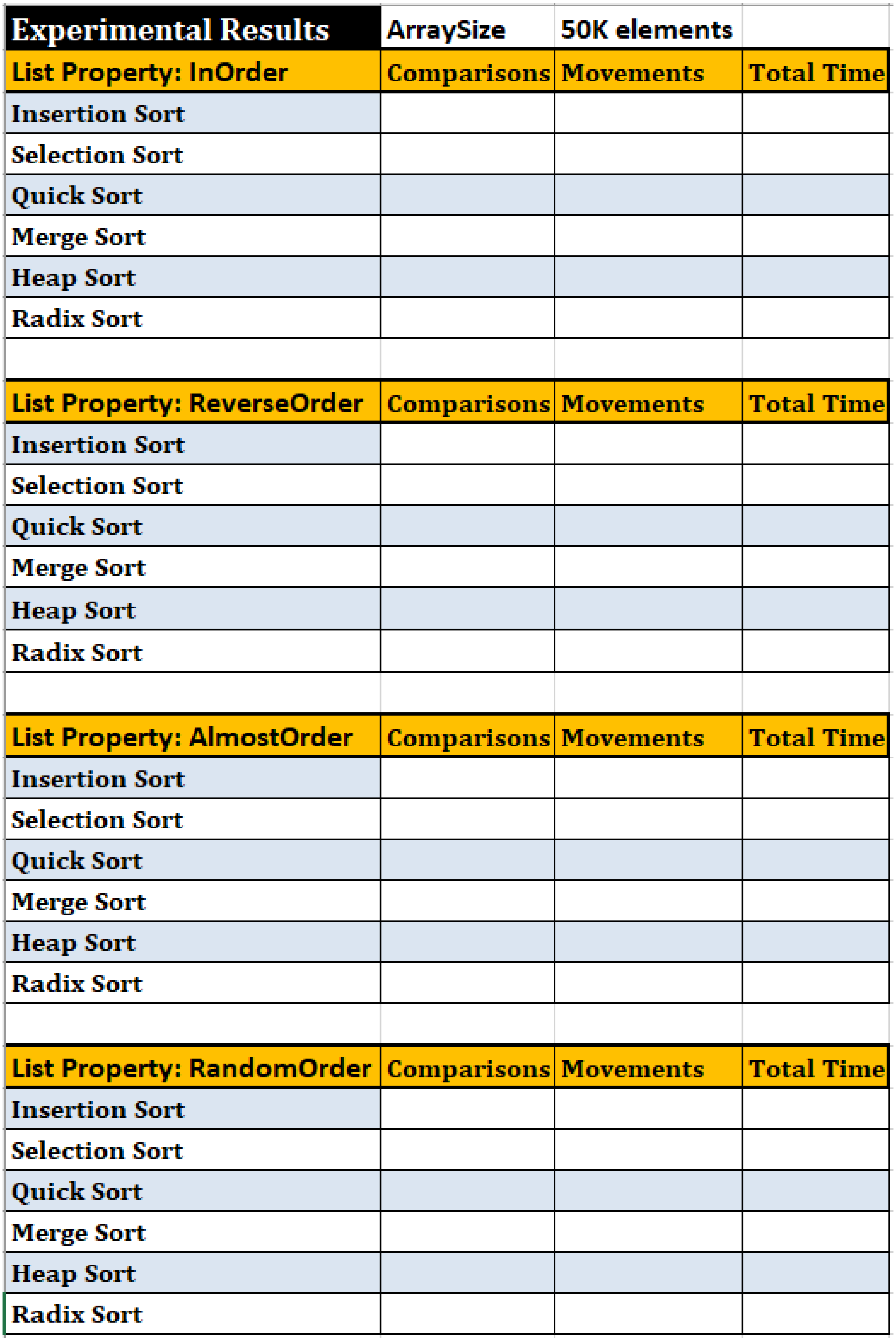
Create and submit a report discussing the analysis at each iteration. Clearly define your approach, challenges, and assessment.

	Project 1 - Sorting Algorithm (Rubric)		Points	Total
1	**Experimental Results**	Comparisons	5	30
	(per data type per sorting algorithm)	Movements	5	30
2	**Analysis Report**			40
	Total Points			100

Caveat:

1. **Student 1**: The quick sort that is provided will run into an infinite loop around 14k elements. So, can we just rewrite the quick sort?
2. **Student 2**: Did you get Stack Overflow? If you did, then it's because the default 1mb of memory storage isn't enough to process more than 14k elements. I had the same problem, and I fixed it by changing the memory storage in my IDE to 3mb under VM argument using the command -Xss3m.
3. **Student 3**: it only has issues during In Order and Reverse Order. I would imagine this is because the pivot selection for this implementation is resulting in there being one side that is very small and one that is very large when the list is sorted.
4. **Student 4:** I changed the pivot to (front + back)/2 instead of front. This is the best for our project in my opinion because it is easy to calculate and solves the issues we are having.
5. **Student 5**: Are you sure it's the sorting algorithm? Me and my buddy found that it was the limits of our printing functions preventing us from checking sort. Once I switched to making my check a for loop of println's, it was able to sort and print at least a billion elements.

Primitive vs Generic Type: Box primitive int to Integer object for algorithms requiring generic



Experimental Results	ArraySize	50K elements	
List Property: InOrder	**Comparisons**	**Movements**	**Total Time**
Insertion Sort			
Selection Sort			
Quick Sort			
Merge Sort			
Heap Sort			
Radix Sort			
List Property: ReverseOrder	**Comparisons**	**Movements**	**Total Time**
Insertion Sort			
Selection Sort			
Quick Sort			
Merge Sort			
Heap Sort			
Radix Sort			
List Property: AlmostOrder	**Comparisons**	**Movements**	**Total Time**
Insertion Sort			
Selection Sort			
Quick Sort			
Merge Sort			
Heap Sort			
Radix Sort			
List Property: RandomOrder	**Comparisons**	**Movements**	**Total Time**
Insertion Sort			
Selection Sort			
Quick Sort			
Merge Sort			
Heap Sort			
Radix Sort			

Non-Graphical User Interface

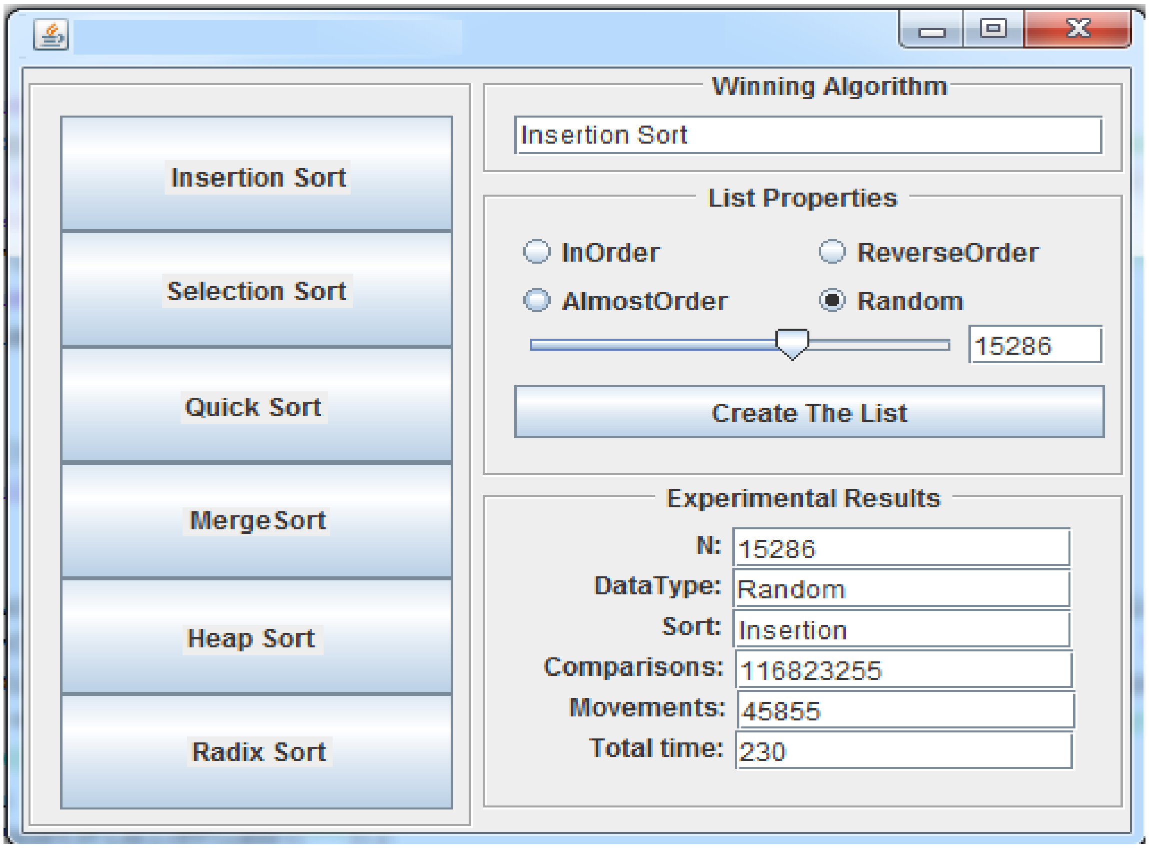
```
1. InOrder
2. ReverseOrder
3. AlmostOrder
4. Random Order
List Properties, select the data type of list you wish to use: 4

1. 5000
2. 15000
3. 50000
Input Size, select the size of list: 2

1. Insertion Sort
2. Selection Sort
3. Quick Sort
4. Merge Sort
5. Heap Sort
6. Radix Sort
Sorting Algorithm, select the sorting algorithm: 1

Experimental Results:
Input Size: 15000
Data Type: Random Order
Sort: Insertion
Comparisons: 116823255
Movements: 45855
Total Time: 230
```

Optional: Graphical User Interface



Winning Algorithm

Insertion Sort

List Properties

○ InOrder ○ ReverseOrder

○ AlmostOrder ● Random

15286

Experimental Results

N: 15286

DataType: Random

Sort: Insertion

Comparisons: 116823255

Movements: 45855

Total time: 230

TASK LIST

You are required to complete the following activities by the deadlines specified and submit the appropriate *deliverables* through eLearning.

ACTIVITY
1. Implement all 6 sorting algorithms to produce the output for each data type applied to each sorting algorithm. For example, insertion sort to be run against all 4 data types (InOrder, ReverseOrder, AlmostOrder, RandomOrder) capturing the experimental results. Similarly, the other 5 algorithms need to do the same. 2. For each data type, determine the best and the worst algorithm based on comparisons, movements, and the time it took to sort the list or any additional criteria you can come up with. 3. Prepare a report discussing 1) how each sorting algorithm works, the best, average and worst-case times for each algorithm. 2) all 24 experimental results and how each data type determined the best and the worst algorithm for that data type
Zip all the Java Files, not the class files, and submit on eLearning under Assignment Section.

GUIDELINES

You will be graded according to the following guidelines:

- You are required to submit files through eLearning by the specified deadline. You can earn a maximum of 100 points.
- If the files do not compile, you will receive a 0 for the program.
- You are graded primarily on the design of your class and the adequate testing of your class. Poor design or inadequate testing will result in loss of points.
- Your files should be adequately commented. Up to -15 points will be deducted for poor indentation and documentation.