Practical 4: Elementary Sorting

What am I doing today?

Today's practical focuses on 3 things:

1. Writing several elementary sorting algorithms
2. Developing a testing framework to assess the performance of your algorithms
3. Summarizing the results

Instructions

Try all the questions. Ask for help from the demonstrators if you get stuck. Look for the helper files in the repo.

Quick Questions

1. **How many compares does insertion sort make on an input array that is *already sorted*?**

Constant	
Logarithmic	
Linear	
Quadratic	

2. **What is a stable sorting algorithm?**

3. **What is an external sorting algorithm?**

A. Algorithm that uses tape or disk during the sort

B. Algorithm that uses main memory during the sort

C. Algorithm that involves swapping

D. Algorithm that are considered ‘in place’

4. **Identify 6 ways of characterizing sorting algorithms?**

Algorithmic Development

Today your mission is to develop a Java class that implements several elementary (and silly) sorting algorithms. The problem we want our algorithms to solve is sort an input array of integers into ascending order and output the resulting array.

Possible steps to follow

1. Create a new java class or use the starter code provided (i.e. Sorts_starter_code.java)
2. Implement the following sorting algorithms as public static functions within your class that take an array of integers and sorts the array, outputting a sorted array of integers:
    a. **Selection sort**
    b. **Insertion Sort**
    c. **A Silly Sort** (either from the list below or of your own making)
3. Create a simple framework for generating input arrays of various sizes (e.g., 10, 1000, 100,000) and then testing the performance over several runs
4. Print the resulting sorted array: Implement a function to print out all elements in the array
5. Time the performance of the previous step on your 3 algorithms and output the execution times for various input sizes (e.g. 10,100,1000) on a **graph**
6. Justify the results of your experiments for the algorithms by proposing the algorithm complexity in big-O notation

Sorting Algorithms PseudoCode

1. Selection Sort

Steps

1. Find the smallest input value (e.g., integer). Swap it with the first input element..
2. Find the second-smallest item. Swap it with the second item.
3. Find the third-smallest item. Swap it with the third item.
4. Repeat finding the next-smallest item, and swapping it into the correct position until the array is sorted.

PseudoCode

2. InsertionSort

Steps

1. The first step involves the comparison of the element in question with its adjacent element.
2. And if at every comparison reveals that the element in question can be inserted at a particular position, then space is created for it by shifting the other elements one position to the right and inserting the element at the suitable position.
3. The above procedure is repeated until all the elements in the array are in their correct position.

PseudoCode

3. Some Silly Algorithms to pick from

For fun or Computer Science comedic fun, implement one of these obscure sorting algorithms and run it through the sequence of steps above:

- **BogoSort: https://en.wikipedia.org/wiki/Bogosort**
  The stupidest sorting algorithm ever created?
- **Stalin Sort: https://www.quora.com/What-is-Stalin-sort**
  The Stalin sort is a joke sort in which elements that are out of order get removed from a list.
- **Slow Sort**: https://en.wikipedia.org/wiki/Slowsort
  "Slow sort is a sorting algorithm. It is of humorous nature and not useful"

***Alternatively, develop your own stupid sorting algorithm**