Warm up – Practice Quiz Questions

1. Which sorting algorithm is used below?
   - 79 58 24 6 19 42
   - 58 79 24 6 19 42
   - 24 58 79 6 19 42
   - 6 24 58 79 19 42
   - 6 19 24 58 79 42
   - 6 19 24 42 58 79

2. Which sorting algorithm is used below?
   - 63 53 87 79 94 19
   - 63 53 87 79 94 19
   - 63 53 87 79 94 19
   - 63 53 87 79 94 19
   - 53 63 87 79 94 19
   - 53 63 87 19 79 94
   - 19 53 63 79 87 94

Go to http://goo.gl/v0ffd

What does "efficiency" mean?

- Measure of how long the algorithm takes to run to completion relative to the number of inputs it is given

Why do we care?

- "Because it helps computer programmers be more green and saves both time and energy which in turn saves money"
- Inefficient algorithms require more calculations/comparisons/memory
  - Wasted time (→ wasted money)
  - Wasted power (→ being less "green")

How do we measure efficiency?

- Big O notation → symbolic execution time
  - *Formula* for the *growth rate* of the time the algorithm takes to complete with respect to the amount of input data.
  - Approximation → no constants, no coefficients
- Linear: $O(n)$
- Quadratic: $O(n^2)$
- Exponential: $O(x^n)$
- Logarithmic: $O(\log n)$
- Constant: $O(1)$

Sorting Efficiency Lab

- Use website to answer lab worksheet questions
- Key Questions to think about:
  - What are the ideal properties of an efficient sorting algorithm?
  - What are the advantages and disadvantages of each sorting algorithm?
  - In what case(s) would you use one over another?

Ideal Properties

- *Stable*: Equal keys (values) aren't reordered.
- Operates *in place*, requiring *O(1) extra space*.
- Worst-case *O(n log n)* key comparisons.
- Worst-case *O(n) swaps* (moves).
- *Adaptive*: Speeds up to *O(n) when data is nearly sorted* or when there are few unique keys.

Advantages/Disadvantages

- Selection Sort
  - Advantages
    - O(1) extra space
    - O(n) swaps
  - Disadvantages
    - Not stable
    - $O(n^2)$ comparisons
    - Not adaptive
- Insertion Sort
  - Advantages
    - Stable
    - O(1) extra space
    - Adaptive: O(n) time when nearly sorted
  - Disadvantages
    - $O(n^2)$ comparisons and swaps

Advantages/Disadvantages

- Merge Sort
  - Advantages
    - Stable
    - O(n log n) time
  - Disadvantages
    - O(n) extra space for arrays
    - Not adaptive

When to use

- Selection Sort
  - In place but not adaptive → real-time, memory limited applications with small data inputs
- Insertion Sort
  - In place and adaptive → good for small, mostly sorted arrays
- Merge Sort
  - Fast, not adaptive → variety of situations when memory is not an issue; better for sequential access