WEEK FOURTEEN

Acknowledgements: Slides created based off material provided by Dr. Michael Raymer and Dr. Travis Doom

ALGORITHMS

- Approach to solving a computational problem
- Algorithms must be:
  - Finite: must terminate eventually
  - Correct: must solve the problem every time
  - Deterministic: made of concrete, computer-executable steps, and produce the same output every time
- Good algorithms are:
  - Bug-free
  - Secure
  - Fast
  - Don’t hog memory

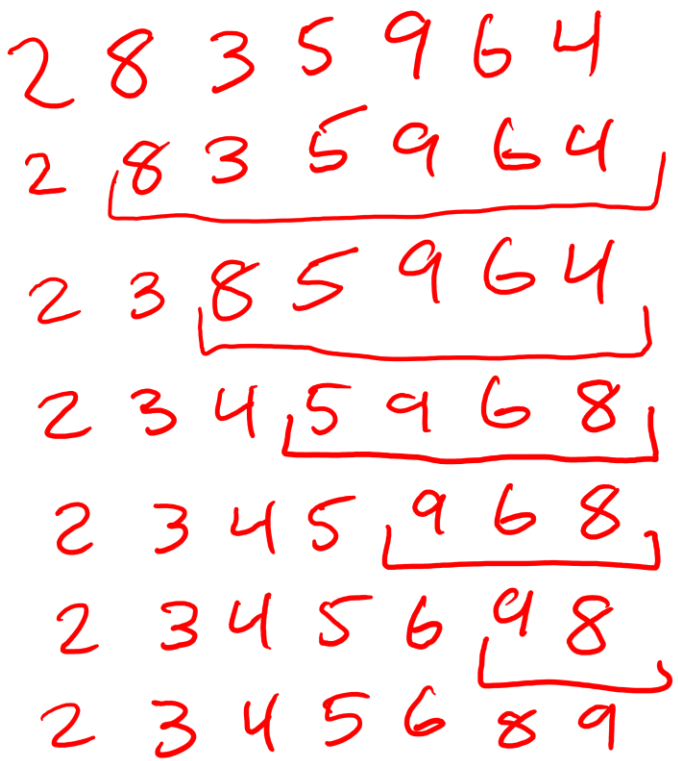
SEARCH ALGORITHMS

- Linear search
  - Step through items one-by-one until the desired item is located
- Binary search
  - Requires a sorted list
  - Split the list in half, check if the value is greater than or less than the central pivot
    - If greater, move to the right side of the list and repeat
    - If less than, move to the left side of the list and repeat

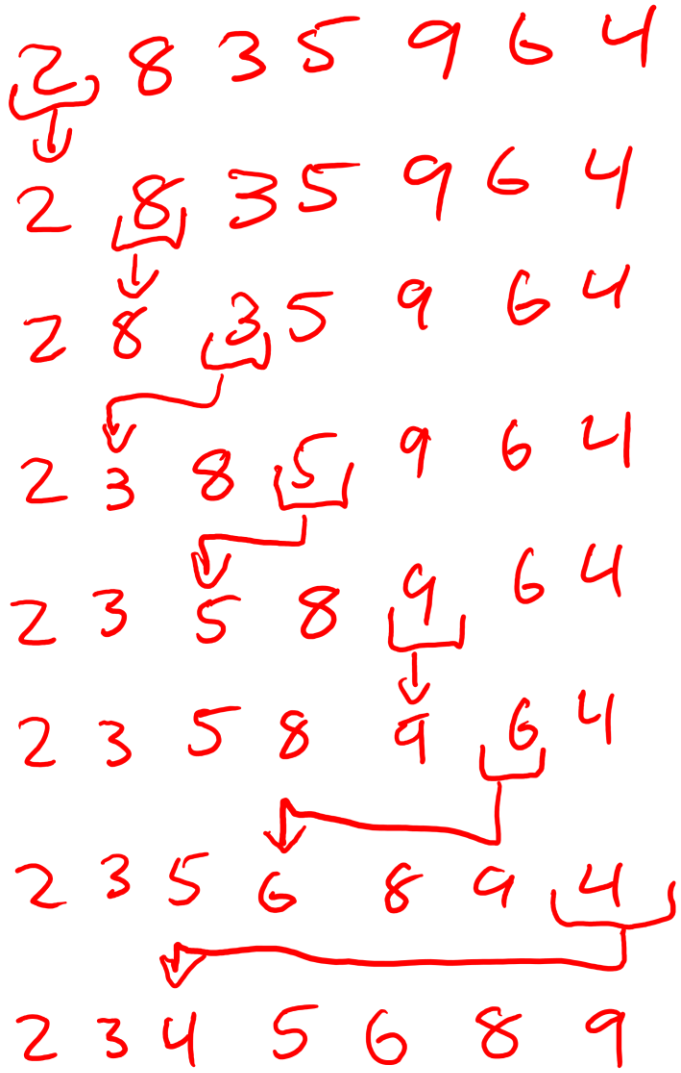
SORTING ALGORITHMS

- Selection sort
  - Remove the min/max item from an unsorted list and add (swap) it to the beginning/end of a new sorted list
- Insertion sort
  - Take each element, in position order, and move it into the appropriate sorted location in a ‘new’ sorted list
- Bubble sort
  - Sort in multiple passes
  - In each pass, successively swap neighboring elements if they are in the wrong natural order
  - Continue passes until all elements are fully ordered

Selection



Insection



Bubble

Pass #1

2 8 3 5 9 6 4

2 8 3 5 9 6 4

2 3 8 5 9 6 4

2 3 5 8 9 6 4

2 3 5 8 9 6 4

2 3 5 8 6 9 4

2 3 5 8 6 4 9

2 3 5 8 6 4 9

2 3 5 8 6 4 9

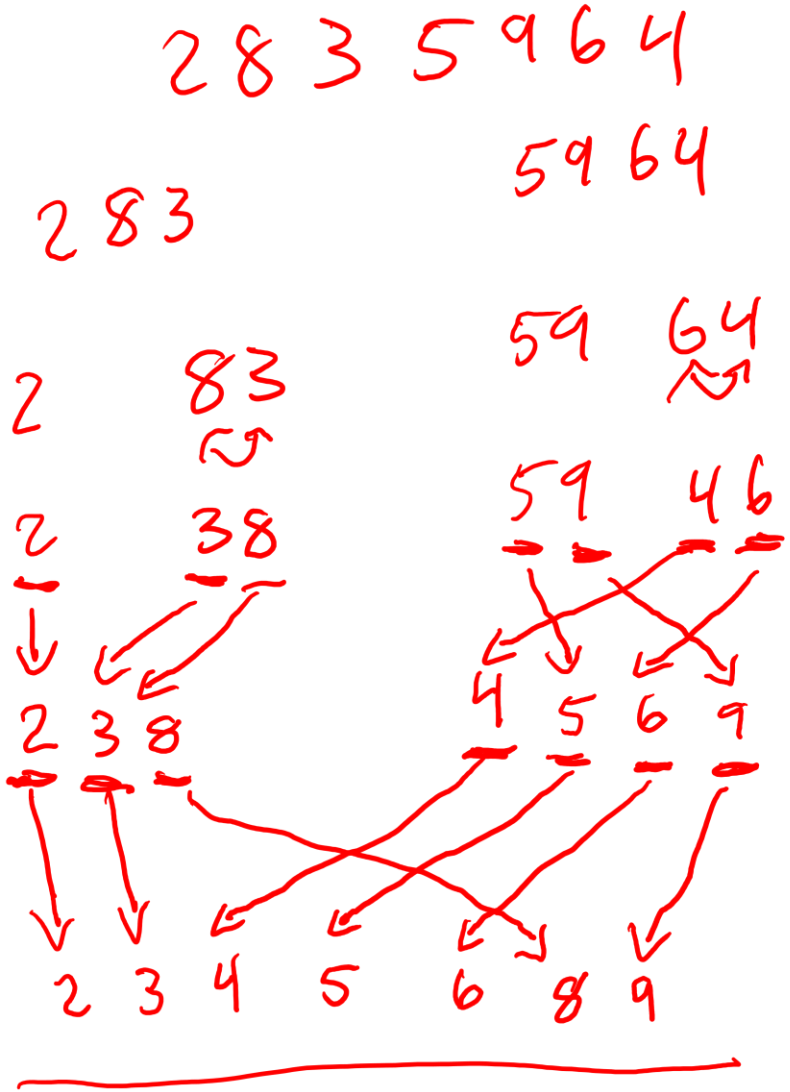
2 3 5 6 8 4 9

2 3 5 6 4 8 9

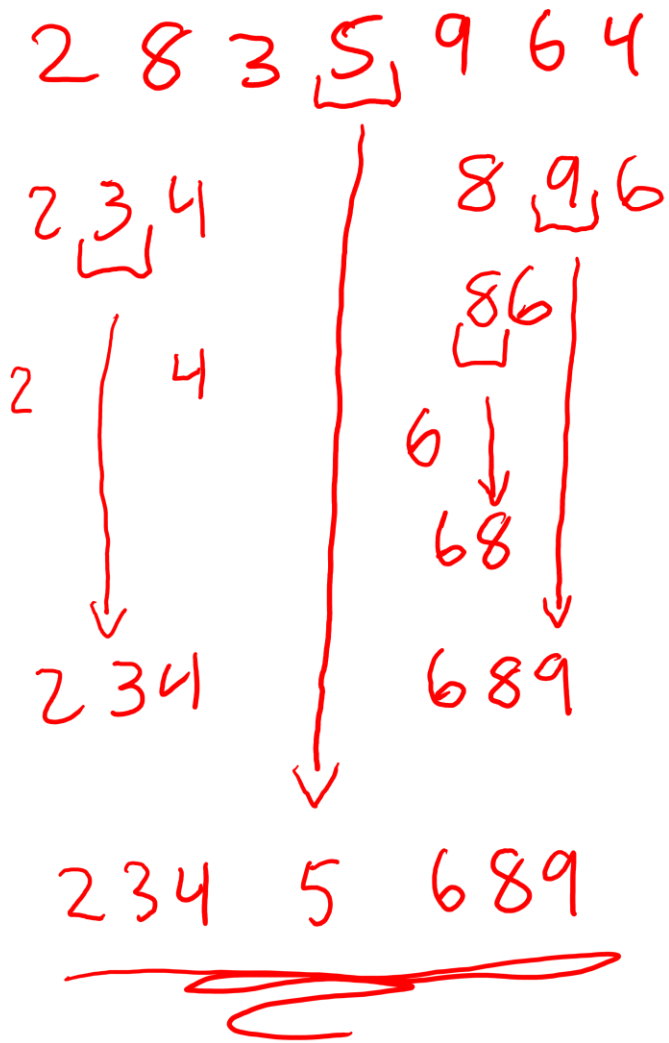
2 3 5 4 6 8 9

2 3 4 5 6 8 9

Merge


2 8 3 5 9 6 4
2 8 3
5 9 6 4
2
8 3
5 9
6 4
2
3 8
5 9
4 6
2 3 8
4 5 6 9
2 3 4 5 6 8 9

Quick


2 8 3 5 9 6 4
2 3 4
8 9 6
8 6
2
4
6
6 8
2 3 4
6 8 9
2 3 4 5 6 8 9

SORTING ALGORITHMS CONTINUED

- Merge sort
  - Usually done recursively
  - Divide array into two halves
  - As base case, sort the two remaining values
  - Upon exiting the recursive call, poll items from each half in order
- Quick sort
  - Pick a pivot
  - Reposition items so all items less than pivot will appear to its left and all greater items will appear on the right
  - Continue on smaller and smaller portions until the list is sorted

BIG-O

- How long an algorithm takes to run in relation to the input
- Function can be simplified by:
  - Removing any constant coefficients
  - Removing all but the highest order term
- Example:
  - $7n^2 + 5^n + 3$ becomes…
  - n^2

COMMON RATES OF GROWTH (BY MAGNITUDE)

- $O(1)$ – constant
- $O(\log n)$ – logarithmic
- $O(n)$ – linear
- $O(n * \log n)$ – log-linear
- $O(n^c)$ – polynomial
  - $O(n^2)$ – quadratic
  - $O(n^3)$ – cubic
  - Etc.
- $O(c^n)$ – exponential
  - $O(2^n)$
  - $O(3^n)$
  - Etc.
- $O(n!)$ – factorial

https://www.desmos.com/calculator/jpskgp3y7a

SORTING ALGORITHMS AND THEIR COMPLEXITY

• Selection	• $O(n^2)$
• Insertion	• $O(n^2)$
• Bubble	• $O(n^2)$
• Merge	• $O(n \log(n))$
• Quick	• $O(n^2)$