Iterative Sorts

Bubble Sort

Bubble sort inspects pairs of adjacent elements, and swaps as necessary which "bubbles" the maximum element to its correct position.

Pseudocode

```
stopIndex = array.length - 1
while stopIndex != 0:
  while i < stop
    if (arr[i] > arr[i+1]):
      swap values at i and i+1
    i++;
  stopIndex--;
```

Note 0.1

One optimization is to make the sort adaptive, meaning it will stop early if necessary. To do this, we have a flag `swapsMade` and if false after an iteration, we can stop early.

Theorem 0.1

Time Complexity

Best Case: O(n) (already sorted)

Average Case: O(n^2)

Worst Case: O(n^2) (reverse sorted array)

Bubble sort is *stable, adaptive, and in-place*

Cocktail Shaker

Iterative sort that performs bubble sort twice in one iteration, to bubble the largest to the end, and the smallest to the front, shaking the list back and forth

The first stage of cocktail shaker sort loops through the array from left to right, swapping values out of order. At the end of the first iteration, the largest number will reside at the end of the array.

The second stage loops backwards from the end, re-swapping when necessary. However, we will start going backwards at the most recently sorted item, since everything past that should be sorted.

Theorem 0.2

Time Complexity:

Best: O(n) (fully sorted array)

Average Case: O(n^2)

Worst: O(n^2) (reverse sorted array)

Cocktail shaker sort is *stable, adaptive, and in-place*