Bubble Sort

Code

```
function bubbleSort(A: number[]) {
    for (let i = A.length; i > 0; i--) {
        let noSwap = true;
        // compare adjacent elements
        // bubbles float to surface
        for (let j = 0; j < i - 1; j++) {
            if (A[j] > A[j + 1]) {
                noSwap = false;
                const tmp = A[j];
                A[j] = A[j + 1];
                A[j + 1] = tmp;
            }
        }
        if (noSwap) break;
    }
    return A;
}
```

Design

- repeatedly step through the array, compare adjacent elements, and swap if they are in the wrong order
- repeat until list is sorted which is confirmed by *no swaps* occurring in the iteration

Runtime Analysis

- worst case is $O(n^2)$
- best case is $O(n)$ if we terminate early after an iteration of no swaps
- average case is $O(n^2)$ (using expectation)