Selection Sort

Code

```
function selectionSort(A: number[]) {
    for (let i = 0; i < A.length; i++) {
        // find index of minimum element
        let min = i;
        for (let j = i; j < A.length; j++) {
            if (A[j] < A[min]) min = j;
        }
        // swap with front of sorted subarray
        if (min != i) {
            const tmp = A[i];
            A[i] = A[min];
            A[min] = tmp;
        }
    }
    return A;
}
```

Design

- the algorithm maintains 2 subarrays in a given array
- in every iteration, the minimum element from the unsorted subarray is added to the end of the sorted subarray

Runtime Analysis

The first iteration will run $n-1$ times, the second will run $n-2$, and so one until 1.

$$
\begin{aligned}
(n-1)+(n-2)+\ldots+1 &= \sum_{i=1}^{n-1}(n-i) \\
&= \frac{(n-1)((n-1)+1)}{2} \\
&= \Theta(n^2)
\end{aligned}
$$

- note that you can use the sum of an arithmetic series formula to show this
- also notice that the runtime analysis is *always* $\theta(n^2)$ for best, worst, and average cases