

CS62: Fall 2025 | Lecture #10 (Selection & Insertion sort) worksheet | Prof. Li

1. Using selection sort, sort the array with elements [12,10,16,11,9,7]. Draw each swap as a new row.

2. Fill in the blanks to implement selectionSort. What is the worst/base case run time? Is it in-place? Is it stable?

```
public class SelectionSort {
    public static <E extends Comparable<E>> void selectionSort(E[] a) {
        int n = _____;
        for (int i = 0; i < n; i++) {
            int min = ____;
            for (int j = _____ ; _____ ; _____) {
                if (a[j].compareTo(a[min]) __ 0) { // > or < or == ?
                    min = _____;
                }
            }
            // do the swap
            _____
            _____
            _____
        }
    }
}
```

3. Using insertion sort, sort the array with elements [12,10,16,11,9,7]. Draw each insertion as a new row.

4. Fill in the blanks to implement insertionSort. What is the invariant after each iteration of i? What is the worst/base case run time? Is it in-place? Is it stable?

```
public class InsertionSort {
    public static <E extends Comparable<E>> void insertionSort(E[] a) {
        int n = ____;
        for (int i = 0; i < n; i++) {
            for (int j = ____; ____; ____) {
                if (a[____].compareTo(a[____]) < 0) {
                    // do the swap
                    _____
                    _____
                    _____
                } else {
                    break;
                }
            }
        }
    }
}
```