

# COMP 1600 Fall 2025

## Lab 14: Selection Sort

### Due by the end of class

The objective of this lab is to implement a simple sorting algorithm. We gave the code for Bubble Sort in class, but you must use the Selection Sort algorithm for this lab.

Your mission is to definition for a function that will sort an array of `int` values using Selection Sort.

### Specification

Create a project called `Lab14`. Add a class called `Sort`. Delete everything inside the file and paste in the code from [Sort.java](#). As you can see, the `main()` method has already been written. All you have to do is write the `sort()` method.

The `main()` method will prompt the user for a length and generate a list of random numbers that long. Then, it will print them out, call the `sort()` method, and print them out again.

You must write the `sort()` method to implement Selection Sort (in ascending order). Recall that the algorithm for Selection Sort looks for the index of the array with the smallest value. Then, you swap the contents of that index with the contents of array index 0. Next, you look for the index of the array with the smallest value, excluding index 0. Then, you swap the contents of that index with array index 1. You continue on, always trying to find the next smallest element and swapping it with the next spot in the array. Thus, you find the final sorted array, element by element, starting at the beginning of the array and moving to the end.

The name Selection Sort comes from the fact that you are scanning the entire array to **select** the next element in the ordering of the array. Here is a pseudocode version of the algorithm, assuming that  $n$  gives the length of the array.

For  $i$  starting at 0 and going up to, but not including,  $n$ :

- Find the smallest element whose index is greater than or equal to  $i$  and less than  $n$
- Swap that element with the element at index  $i$

Hint: You will need two nested loops. The outer loop iterates over the current position you are trying to fill. The inner loop finds the smallest element in the remaining part of the array.

## Sample Output

The provided `main()` function will generate a list of random numbers. Here is an example of sample output for input length 20. User input is shown in **green**. Your output will probably not match, since the values are generated randomly.

```
How many values would you like to generate? 20

The numbers are:
3 570 768 361 122 916 582 594 479 967 290 125 699 676 115 955 366 182 654 403

The numbers sorted are:
3 115 122 125 182 290 361 366 403 479 570 582 594 654 676 699 768 916 955 967
```

You should test your code several times to make sure it's right. Because the numbers are random, there's a chance that an incorrect sort could sort some lists of numbers correctly but fail to sort others. If your list of numbers is not correctly sorted, the `main()` method will print a warning.

## Turn In

Turn in your code by uploading `Sort.java` from the `Lab14\src` folder wherever you created your project to [Brightspace](#). **Do not** upload the entire project. I only want the `Sort.java` file.

All work must be done individually. Never look at someone else's code. Please refer to the course policies if you have any questions about academic integrity. If you have trouble with the assignment, I am always available for assistance.