Topic: Sorting Algorithms – Bubble Sort and Insertion Sort

Activity Guidelines

Group Size: 3

Method of Assigning Students: Count the number of students in the class, divide by 3, count off from 1 to the quotient, and group identical numbers.

Materials:

- ✓ Handout (one copy per group) with questions to be answered at the end of the session

Roles:

Coordinator/Leader: Clarifies goals and objectives, allocates roles for each team member and divides the tasks within the group.

Monitor/Evaluator: Person designed to evaluate the different ideas to approach the problem and make an accurate judgment of the most beneficial option.

Implementer: Person in charge to transform discussions and ideas into a technical solution for the given problem.

Individual Accountability: Each team member gets assigned a specific role in order to ensure every student within a team participates and contributes to reach a solution for each problem presented in the activity.

Activity Summary

- Students are provided with Bubble Sort and Insertion Sort algorithms for the assignment.
- Each team is assigned an array of numbers set at random order.
    - **a.** Team #1 is in charge of sorting the array with Bubble Sort in descending order.
    - **b.** Team #2 is in charge of sorting the array with Bubble Sort in ascending order.
    - **c.** Team #3 is in charge of sorting the array with Insertion Sort in descending order.
    - **d.** Team #4 is in charge of sorting the array with Insertion Sort in ascending order.

REFERENCE BUBBLE SORT: http://www.algolist.net/Algorithms/Sorting/Bubble_sort
REFERENCE INSERTION SORT: http://www.algolist.net/Algorithms/Sorting/Insertion_sort

ELEMENTARY DATA STRUCTURES

PEER SESSION

Bubble Sort

```
public void bubbleSort(int[] arr) {
	boolean swapped = true;
	int j = 0;
	int tmp;
	while (swapped) {
		swapped = false;
		j++;
		for (int i = 0; i < arr.length - j; i++)
{
			if (arr[i] > arr[i + 1]) {
				tmp = arr[i];
				arr[i] = arr[i + 1];
				arr[i + 1] = tmp;
				swapped = true;
			}
		}
	}
}
```

Insertion Sort

```
void insertionSort(int[] arr) {
	int i, j, newValue;
	for (i = 1; i < arr.length; i++) {
		newValue = arr[i];
		j = i;
		while (j > 0 && arr[j - 1] > newValue) {
			arr[j] = arr[j - 1];
			j--;
		}
		arr[j] = newValue;
	}
}
```

Bubble Sort

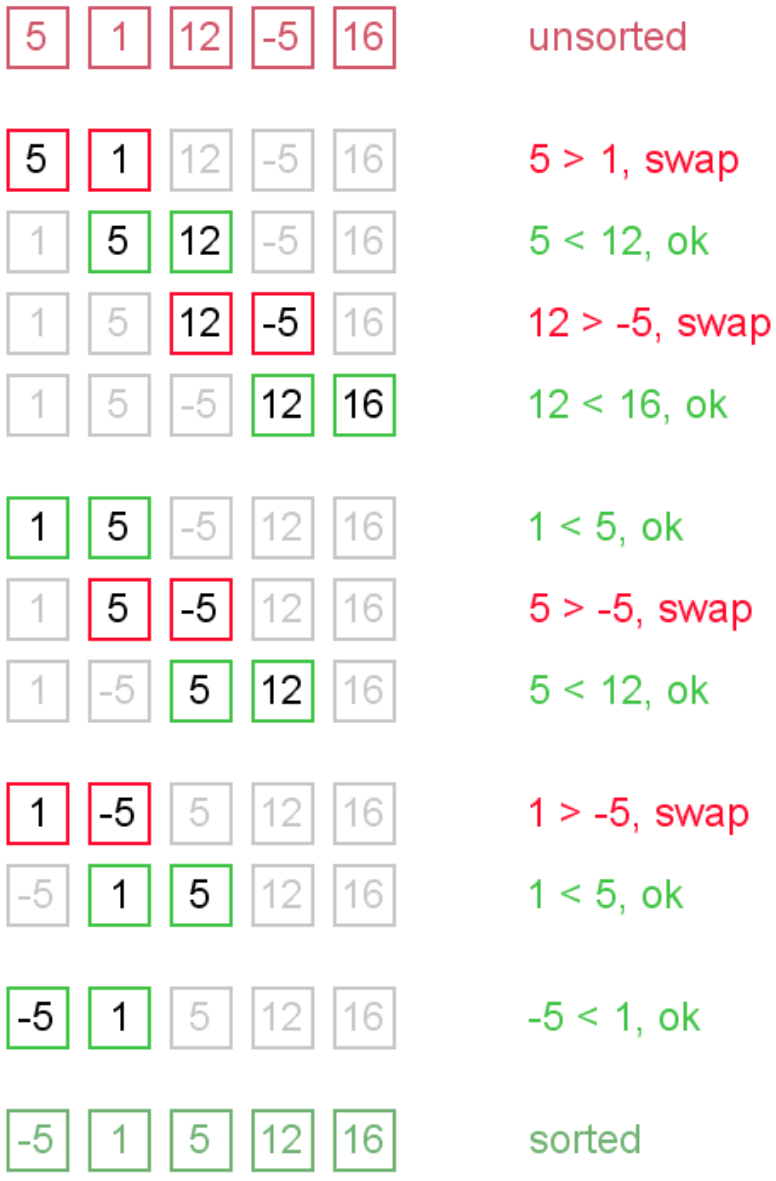
Bubble sort is a simple and well-known sorting algorithm. It is used in practice once in a blue moon and its main application is to make an introduction to the sorting algorithms. Bubble sort belongs to $O(n^2)$ sorting algorithms, which makes it quite inefficient for sorting large data volumes. Bubble sort is **stable** and **adaptive**.

Algorithm

1. Compare each pair of adjacent elements from the beginning of an array and, if they are in reversed order, swap them.
2. If at least one swap has been done, repeat step 1.

You can imagine that on every step big bubbles float to the surface and stay there. At the step, when no bubble moves, sorting stops. Let us see an example of sorting an array to make the idea of bubble sort clearer.

Example. Sort {5, 1, 12, -5, 16} using bubble sort.



Insertion Sort

Insertion sort belongs to the $O(n^2)$ sorting algorithms. Unlike many sorting algorithms with quadratic complexity, it is actually applied in practice for sorting small arrays of data. For instance, it is used to improve quicksort routine. Some sources notice, that people use same algorithm ordering items, for example, hand of cards.

Example. Sort {7, -5, 2, 16, 4} using insertion sort.