Lab5: Prelab

Sorting Algorithms

Sorting… Why?

Broadly speaking, sorting is a procedure to go from the original state of a list to a permutation of that state. A large amount of information we have to deal on the daily basis is in a random order. So we use sorting in order to:

- Better understand said information(structure)
- Exploit the knowledge of the order of the structure(lists, arrays) after sorting

Goals:

- Be able to visualize procedures involved in various sorting algorithms
- Implement sorting algorithms in C.
- Strengthen concepts of control structures(nested loops) and understand multiple approaches to solve same problem(3 ways to sort arrays).

Description:

In the upcoming lab you shall implement three sorting algorithms in C:

- Bubble Sort
- Insertion Sort
- Selection Sort

Brief Intro to Different Sorting Algorithms:

- **Bubble Sort**:
  Sometimes referred as sinking sort, it's a simple sorting algorithm that loops through the list to be sorted, and repeatedly compares each pair of adjacent items and swaps them if they are in the wrong order.

  Below is the animated visualization of bubble sort:
  http://www.cs.armstrong.edu/liang/animation/web/BubbleSort.html

  Pseudo Implementation:

```
procedure bubblesort( var arr as array of length N )
    for i ← 1 to N
        for j ← 0 to N - 1
            if arr[j] > arr[j + 1]
                swap( arr[j], arr[j + 1] )
end procedure
```

- **Insertion Sort:**
  It builds the final sorted array (or list) by inserting one item to the currently sorted list once a time.

  Below is the animated visualization of how insertion sort works:
  http://cs.armstrong.edu/liang/animation/web/InsertionSort.html

  Pseudo Implementation:

```
proc insertion_sort(integer array a of length N)
      for i ← 1 to N
                 j ← i - 1
           while(j >= 0 and a[j] > a[j + 1])
                 swap(a[j], a[j + 1])
                       j ← j - 1
           end while
      end for
end proc
```

- **Selection Sort:**
  This is a sorting algorithm by specifically an in-place comparison sort.

  Below is the animated visualization of selection sort:
  http://www.cs.armstrong.edu/liang/animation/web/SelectionSort.html

  Pseudo Implementation:

```
proc selection_sort(integer array a of length N)
    for i ← 0 to N - 1
        maxIndex = i
        for j ← (i + 1) to (N - 1)
            if a[j] > a[maxIndex]
                maxIndex = j
        end for
        swap(a[i], a[maxIndex])
    end for
end proc
```