Assignment 1121

Having (presumably) assimilated the conceptual mechanics of the sequential search, binary search, insertion sort, and quicksort algorithms, as well as the general approaches for iteration and recursion, it's time to try writing out iterative and recursive algorithms yourself, either in JavaScript or in pseudocode.

Not for Submission

As has been mentioned, we are currently on Chapter 5 of the Brookshear book.

For Submission

Implement the following algorithms. Use the JavaScript scratch page for the iteration algorithms, and use pseudocode for the recursive one.

Iteration

- **Exponent of a number** — Use Input 1 to accept a number, and use Input 2 to accept the power to which that number should be raised. Provide an alert that displays the calculated exponent. For example, if Input 1 holds “3” and Input 2 holds “4,” then the answer is $3^4$.
- **Highest/lowest number finder** — Use Input 1 to accept a comma-separated list of numbers. Provide an alert that displays the highest and lowest numbers in that list.
- **Sequential search** — Use Input 1 to accept a comma-separated list of items, and use Input 2 to take the item being searched. Provide an alert that says “yes” if the target is found and “no” if it isn’t found.

Recursion

- **Descendant counter (pseudocode)** — Given a person and that person’s family tree, count all of his or her descendants, across all generations. Provide this one in pseudocode.

Extra Credit

You will get extra credit if you also implement one of these algorithms in JavaScript, using the JavaScript scratch page:

- **Insertion sort (numbers)** — Use Input 1 to accept a comma-separated list of numbers. Provide an alert that shows the sorted version of this list.
- **Binary search** — Use Input 1 to accept a comma-separated list of items, and use Input 2 to take the item being searched. Provide an alert that says “yes” if the target is found and “no” if it isn’t found. Note that this works exactly like sequential search on the outside; it’s the internal work done by the program that is different.
- **Quicksort (numbers)** — Use Input 1 to accept a comma-separated list of numbers. Provide an alert that shows the sorted version of this list. Note that this works exactly like insertion sort on the outside; it’s the internal work done by the program that is different.