COSC 311, Project #2

A laboratory experiment: Is an unsorted list really O(n) to search?

Distributed: 9/30/2015 **Due**: 10/14/2015 (2 weeks)

Precis: Run an experiment to experimentally justify the theoretical result that to search an unordered list takes `O(n)` time.

You will implement an unsorted list in two ways:
(1) As a linked list with direct access to the front of the list,
(2) As an ArrayList

You will run the following experiment on each data structure. Each run will collect measurements as given below.

A run:
Create an unsorted list of n elements of random generated `int` values in the range `[0 .. n-1]`

Do 100 retrievals (or 1000 if necessary) of a random key value (key is in the range `[0 .. n-1]`

The minimum value of n is 1000. n will range up to very big. The size of "very big" will depend on the CPU speed of your system.

Example: `n ε [10^3, 10^5, 10^7, 10^9, 10^11]`

Collect the following measurements:
Size of n, total time for 100 retrievals, average time for retrievals

Data structure	n	total time	average time
Linked list	1000	16	0.016
Linked list	10,000	149	1.49
…	…	…	…
ArrayList	…	…	…
ArrayList	…	…	…

Results: You will provide the results as a table and as a graph for each data structure (two tables with two graphs). The graph and table both must be nicely formatted using some computer tool such as MS Excel.

Follow good scientific communication principles: properly label axes, put independent variable on abscissa, etc.

Using Excel, or other program, fit three curves: O(1), O(n), O(n^2).

Conclusion: Give a one to 5 statement conclusion that says whether your results support or do not support the thesis. Justify your statement using actual experimental results.

Notes:
The table of results does not have to be automatically generated from your experimental runs.

The values of n should be well distributed to obtain convincing graph.

Turn in:
Hard copy of code with the last four digits of your E# -- no name!
Screen shot showing the successful execution
Tables, Graphs
Conclusion

Grade based on:
Satisfying specifications
Code elegance
Code readability
Experiment report (Results and Conclusion)