York University
EECS 2011Z Winter 2015 – Problem Set 3
Instructor: James Elder

This problem set will not be graded, but will help you consolidate the material from the second half of the course and prepare for the final exam. You are free to work together on these if you prefer.

1. Choosing a data structure

   State in one or two words the simplest ADT and implementation we have discussed that would meet each requirement.

   (a) O(1) time removal of the most recently added element
   ADT: Implementation:

   (b) O(1) average time addition, removal, access and modification of (key, value) pairs with unique keys
   ADT: Implementation:

   (c) O(1) time insertion and removal when you are given the position
   ADT: Implementation:

   (d) O(1) time index-based access and modification and amortized O(1) addition of elements
   ADT: Implementation:

   (e) O(log n) time insertion of (key, value) entries and O(log n) removal of entry with smallest key
   ADT: Implementation:

   (f) O(1) time removal of the least recently added element
   ADT: Implementation:

2. Binary Search Trees
   Insert, into an empty binary search tree, entries with keys 30, 40, 24, 58, 48, 26, 11, 13 (in this order). Draw the tree after each insertion.

3. AVL Trees
   Insert, into an empty binary search tree, entries with keys 62, 44, 78, 17, 50, 88, 48, 54 (in this order). Now draw the AVL tree resulting from the removal of the entry with key 62.

4. Splay Trees
   Perform the following sequence of operations in an initially empty splay tree and draw the tree after each set of operations.

   (a) Insert keys 0, 2, 4, 6, 8, 10, 12, 14, 16, 18, in this order.

   (b) Search for keys 1, 3, 5, 7, 9, 11, 13, 15, 17, 19, in this order.

   (c) Delete keys 0, 2, 4, 6, 8, 10, 12, 14, 16, 18, in this order.

5. Comparison Sorts
   Of the $n!$ possible inputs to a given comparison-based sorting algorithm, what is the absolute maximum number of inputs that could be sorted with just n comparisons?

6. Comparison Sorts
   Give an example input list for which merge-sort and heap-sort take $\mathcal{O}(n \log n)$ time, but for which insertion sort takes $\mathcal{O}(n)$ time. What if the list is reversed?

7. Stack-Based Quicksort
   Describe in pseudocode a non-recursive version of the quick-sort algorithm that explicitly uses a stack.

8. Linear Sorts
Given an array of n integers, each in the range $[0, n^2 - 1]$, describe a simple method for sorting the array in $\mathcal{O}(n)$ time.

9. Topological Sorts
Suppose that you wish to take a sequence of language courses with the following prerequisites:

Course	Prerequisite
LA15	none
LA16	LA15
LA22	none
LA31	LA15
LA32	LA16, LA31
LA126	LA22, LA32
LA127	LA16
LA141	LA22, LA16
LA169	LA32

(a) Draw a directed graph that represents these dependencies.

(b) Use the topological sorting algorithm to compute a feasible sequence.

10. DFS and BFS
Let G be an undirected graph whose vertices are labelled by the integers 1 through 8, and having the following edges:

Vertex	Edges
1	2, 3, 4
2	1, 3, 4
3	1, 2, 4
4	1, 2, 3, 6
5	6, 7, 8
6	4, 5, 7
7	5, 6, 8
8	5, 7

Assume that, in a traversal of G, the adjacent vertices of a given vertex are returned in the order above.

(a) Draw G.

(b) Give the sequence of vertices of G visited using a DFS traversal starting at vertex 1.

(c) Give the sequence of vertices visited using a BFS traversal starting at vertex 1.