EPGY 2013 - Artificial Intelligence Review Quiz 1/3: Data Structures, Algorithms, Big O

Name ____________________________________

Basic Searches

1. Given a a random, unsorted ArrayList of integers, describe in pseudocode your most efficient algorithm to locate an element, or display an error if that element not found. What if instead of an ArrayList, you have a LinkedList?

2. Using the algorithm you described, what is the worst case scenario of the number of integers you must access if there are 5 integers in the list? 10 integers in the list? 20 integers? 100 integers? 1 billion integers? Qualitatively and/or quantitatively describe this trend. Plotting a few values of this out may help.

3. Given a sorted LinkedList of integers, describe in pseudocode an efficient algorithm to locate an element.

4. How is this similar or different to your result to the unsorted lists? Does the sortedness of the LinkedList matter? Why or why not?

5. Given a sorted ArrayList of integers, describe in pseudocode an efficient algorithm to locate an element.

6. Again, how does this compare to your other algorithms? Does the sortedness of an ArrayList matter? Why or why not?

7. Now, given a sorted binary tree of integers, describe in pseudocode an efficient algorithm to locate an element.

8. Give the worst case number of values you must access for 5 integers. 10 integers? 20 integers? 100? 1 billion? Qualitatively and/or quantitatively describe this trend.

9. What is the relation between searching a binary tree and an efficient searching of a sorted vector?

10. Given a HashSet, describe in pseudocode an efficient algorithm to locate an element.

11. Give the worst case number of values you must access for 5 integers. 10 integers? 20 integers? 100? 1 billion? Qualitatively and/or quantitatively describe this trend.

12. How does HashSet achieve this trend?

13. (Extension) (Challenge) Given an undirected, connected graph (so all nodes are accessible from any starting node), describe in pseudocode an efficient algorithm to locate an element. Make sure you don't visit a node more times than necessary.

Finding the Maximum

(Note: A data structure that supports finding and removing the maximum element efficiently is usually called, as we saw in class, a priority queue.)

1. Now, given an unsorted ArrayList of integers, describe in pseudocode an algorithm for finding the largest element, and removing it. Remember that an ArrayList is backed by an array, so after removing an element, you must iteratively fill in the gap that results.

2. Given a sorted ArrayList of integers, describe in pseudocode an algorithm for finding the largest element, and removing it.

3. Given a HashSet of integers, describe in pseudocode an algorithm for finding the largest element, and removing it.

4. For each of the above, what is the worst case number of operations (where an operation is either an access of an element, a removal of an element, or the moving or swapping of an element) to find the maximum element if there are 5 integers. 10 integers? 20? 100? 1 billion? Describe these trends quantitatively and/or qualitatively.

5. (Extension) (Challenge) A binary heap is a data structure that has the following characteristics: (1) It is a binary tree (not sorted), and (2) at each node, all children of that node has elements that are less than the element at that node. When an element is added to a binary heap, it is always added to the bottom-right. Then, it “bubbles up” until it is smaller that its parent. Removal of a node is similar. After a node is removed, the larger of the node’s children bubbles up to take its place. That leaves a removed node where that child bubbled up from, so we can again bubble up another node from below there. This continues until the tree has is once again a heap. So, given a heap, what is the worst case number of operations to find and remove the max if there are 5 integers? 10? 20? 100? 1 billion?

Sorting

1. Given an unsorted ArrayList of integers, describe in pseudocode an algorithm to sort the integers.

2. What’s the worst case number of operations if you have 5 integers? 10? 20? 50? 100? 1 billion?

3. Do it again. Think harder. Devise another algorithm that does better. Ask around if you’re stuck.

4. (Challenge) Given an unsorted LinkedList of integers, describe in pseudocode an algorithm to sort the integers.

5. (Challenge) What’s the worst case number of operations if you have 5 integers? 10? 20? 50? 100? 1 billion?

6. (Challenge) Given a heap, describe in pseudocode an algorithm to sort the integers.

7. Bubblesort is well-known as one of the first sorting algorithms that first-time programmers tend to use to sort a list before they have learned about Big-O and algorithmic complexity. Bubblesort works by iterating through a list, and swapping pairs if they are out of order. After one pass, more passes are taken, swapping adjacent values each time, until eventually the list is sorted. This is also perhaps the easiest sort to implement. Implement it. In your submission, show it sorting a sorted list of 10 integers, a random list of 10 integers, and a reverse sorted list of 10 integers. What is its Big-O?

8. The following functions take an arraylist of length n. What is the Big-O of the number of operations needed (comparisons, additions, multiplications, etc) to finish the function?

```
private static int foo(ArrayList<Integer> array) {

      int temp;
      for (int i = 2; i < array.size(); i++) {
            if (i < array.get(i)) {
                  temp = array.get(i);
                  array.set(i, array.get(i - 2));
                  array.set(i - 2, temp / 2);
                  i -= 1;
            }
      }
      return temp;
}

private static int bar(ArrayList<Integer> array) {
      if (array.size() < 2)
            return 1;
      else
            return baz(array, array.size());
}

private static int baz(ArrayList<Integer> array, int k) {
      return bar(array.sublist(0,k/2)) + baz(array, k-1);
}
```