

Topics List for Exam 2

- Queues
 - Know how we implemented our own queue.
 - Know how we specified our own Queue interface.
 - know what add(), poll(), peek(), isEmpty(), and size() do.
- Priority Queues
 - Know how to create an empty PriorityQueue.
 - How to create a PriorityQueue using a custom Comparator.
 - How to implement a class that implements the Comparator<E> interface.
 - Know what add(), poll(), peek(), size() and isEmpty() do, and be able to use them to write a method.
 - Be able to solve a problem using a priority queue.
 - Our own implementation:
 - To be able to identify whether or not an array represents a legal binary min heap.
 - Understand the implementation we used (binary min heap.)
 - Be able to run the insertElement() and removeMin() operations on a heap.
 - Know the running times for all of the priority queue methods.
- Sorting
 - Know how to run the $O(n^2)$ time sorting algorithms that we spoke about.
 - bubble sort
 - Insertion sort
 - Know how to run the subroutines of the $O(n \log n)$ time sorting algorithms
 - Merge sort (you have to know how to run merge())
 - Quicksort (you have to know how to run partition())
 - Know how to run the full merge sort and quicksort algorithms.
 - Know the running times for all of the sorting algorithms discussed in class.
 - Know the best and worst cases (and best and worst case times) for each sorting algorithm.
- Sets
 - Know how to create a TreeSet and HashSet.
 - Know that the iterator for TreeSet gives you back the elements in a sorted order.
 - Know what the operations add(), contains(), and remove() do, and be able to use them to write a method to solve a problem.
 - Know how to iterate through a set using both an exposed iterator (similar to Lists) and the for-each loop.
 - To know which mathematical set operations correspond to removeAll(), addAll(), retainAll(), and what these operations do.

- Know how to solve a problem with a set (e.g.
- Maps
 - Know how to create a TreeMap and HashMap.
 - Know how to iterate through a map (using keySet() combined with get(); and entrySet())
 - Know how to use get() and put() and containsKey().
 - Know how to solve a problem with a map (e.g. counting occurrences)
- Binary Search Trees
 - Understand why binary search trees are useful in implementing sets and maps
 - Know how to write a method that searches a BST for a particular element.
 - Know how to write a method that inserts an element into a BST
 - Know how to write preorder/inorder/postorder traversals
 - Know how to traverse a BST to solve a particular problem (e.g. summing all numbers in a BST).