Topics List for Exam 2

- Queues
    - Know how to create an empty queue
    - know what push(), pop(), empty(), and size() do.

- Priority Queues
    - Know how to create an empty priority_queue.
    - Know what push(), pop(), top(), size() and empty() do, and be able to use them to write a function
    - Be able to solve a problem using a priority queue.
    - Our own implementation:
        - Understand the implementation we used (binary max heap.)
        - Be able to run the insertElement() and removeMax() operations on a heap.
        - Know the running times for all of the priority_queue functions.
- Sorting
    - Know how to run at least one of the $O(n^2)$ time sorting algorithms
        - bubble sort
        - Insertion sort
        - Selection sort
    - Know how to run at least one of the O(n log n) time sorting algorithms
        - Merge sort
        - Quicksort
    - Know the running times for all of the sorting algorithms discussed in class

- Sets
    - Know how to create an empty set.
    - Know what the operations insert(), erase(), find(), and count() do, and be able to use them to write a function to solve a problem. (You do **NOT** have to worry about the return type of insert of know how to use it. You must know how to use all other return values)
    - Know how to iterate through a set using set<T>::iterator and const_iterator.

- Maps
    - Know how to create an empty map.
    - Know what the operations insert(), erase(), find(), count(), and operator[] do, and be able to use them to write a function to solve a problem. (Just as with sets, you need not worry about the return type and value for insert()).
    - Know how to iterate through a map using map<T1, T2>::iterator and const_iterator, and know how to handle each element that comes from the iterator (the key is the first component of the pair, and the value is the second component of the pair).

- Binary Search Trees
    - How to create TreeNodes.
    - Know how to write a function to insert a new item into a binary search tree.
    - Know how to search for an item in a binary search tree.
    - Know how to visit all nodes in a binary search tree and perform some operation (like printing out all nodes, adding up all nodes, etc.)