Review sheet for final exam

The format of the final exam will be similar to that of the midterm. You will not be tested directly on data structures covered in the midterm, but if you are asked to select the best structure for a particular situation, you should consider all data structures discussed in the course (including those from the first half).

Study the algorithms we have discussed.

The exam will be closed book/closed notes.

Topics

The topics that the final exam will concentrate on are:

1. **C++/STL**
   You may have to write small snippets of code. These must be syntactically/semantically correct to receive full credit.

2. **Complexity analysis**
   You should be able to evaluate an algorithm's efficiency (time and space) using big-Oh notation. In addition, you are expected to know the time complexity of various data structures' basic operations (e.g. search, insert, etc) or deduce the complexity of new operations.

3. **Sorting**
   Insertion Sort, Heapsort, Quicksort, Mergesort, Counting Sort. Lower bounds for comparison sorts. Stable sorting algorithms. Running times. The importance of selecting a good pivot in Quicksort. Using Quicksort to solve the selection problem. Using Quicksort in conjunction with Insertion Sort. Choosing the best sorting algorithm for a particular situation (e.g. a situation that requires a stable algorithm, one where the input is partially sorted, etc.)

4. **Tries**
   What are tries, how are they organized? Running times and applications. Advantages and disadvantages of compressed tries. PATRICIA trees.

5. **B-trees**
   In what situations are B-trees used? Insert/Delete, running times. What are B* and B+ trees? When do we use them?

6. **Hashing**
   Definition, applications. How do we choose a hash function? How can collisions be resolved? Chaining and open addressing (linear probing, quadratic probing, double hashing). When is rehashing necessary? Apply and compare methods for resolving collisions. Running times.

7. **Disjoint Sets**
   Definition, applications, implementations, operations and running times.

8. **Graphs**
   Definitions, representation, depth-first and breadth-first traversals. How can we use breadth-first traversal to solve the Shortest Path problem? Dijkstra's algorithm for single-source shortest paths. Topological Sort (as an application of depth-first traversal). You should be able to apply the algorithm and show the time stamps. Kruskal's algorithm for the Minimum Spanning Tree problem. What data structures are used in these algorithms' implementations?

   Graph-related terms you'll be expected to know: degree of a vertex, adjacent vertices, edges incident to a vertex, paths, weighted graphs, complete graphs, cycles, connected graphs, directed/undirected graphs. You should be able to answer short questions like "What is the number of edges in a complete graph?", "Why is the size of an adjacency list representation $O(|V| + |E|)$?".

Things NOT on the exam

Top-down splaying. Insert/Delete in B* and B+ trees. Universal Hashing.

Format of the exam

The final exam will be similar to the midterm. There will be some questions that require short answers, some questions where you have to select the best data structure for the job, questions where you'll have to apply some algorithm (e.g. an MST algorithm, an insert/delete/search on some data structure, etc), questions where you may have to write some code or pseudocode, etc.