Final exam study guide

The format would be similar to the other two exams. It will include a combination of multiple choice, short answers and long answers such as writing codes.

There will be:

15 to 25 multiple choice questions – 25 points

15 to 20 short answers --- 45 points

4 programs ----35 points

Some of the questions are numeric types – such as:

Shortest path Dijkstra’s algorithm:

* Show adjacency matrix
* Show step by step of the algorithm modifying the table to get to the final answer

Minimum spanning tree:

* Use prim’s algorithm to find the minimum spanning tree
* Use Krauskal’s algorithm to find the minimum spanning tree

BFS and DFS:

* Given a graph, show the steps of visitation using DFS
* Given a graph, show the steps of visitation using BFS
* The pseudo code for both one

Huffman Tree:

* Given a sequence of characters used in a text file, show the Huffman tree
* Show how a given message will be encoded and decoded

AVL Tree:

* Given a sequence of values, generate an AVL tree
* Determine the rotations needed at each step
* Given a BST determine if it is an AVL

Heap Structure:

* Given a list of items with associated priorities, show how you can create a heap binary tree by adding each item one by one
* Given a binary heap tree, how do you place the values in an array
* Given a binary tree, show how you can make it into a heap
* Show how heapsort works

Search Trees:

* Given a BST, show post order, preorder and in order traversals
* Given a preorder traversal for BST, show its post order traversal
* 2-3, 2-4 and B-Trees

Stack Applications:

* Show if an expression has a balanced delimiter using stack
* Convert an infix into postfix
* Evaluate an expression using stack

Queue:

* Show the condition where a circular buffer is empty using an unused buffer location
* Different strategies to find out if a circular queue is empty or full

Sort algorithm:

* Quicksort
* Mergesort
* Radix sort
* Heapsort

Hashing:

* Open addressing techniques: Linear probing, quadratic probing
* Separate chaining
* Rehashing (what is the process of rehashing and when do we need to rehash)
* Load factor- how to calculate the load factor and why we use it
* Cuckoo Hashing
* Hopscotch Hashing

Disjoint Sets:

* Given disjoint sets, how do you do union by height and union by size
* Given an algorithm such as Krauskal’s algorithm show how find/union can be used to detect cycles

Short codes:

* Find number of nodes in a BST
* Find max value in a BST
* Write the code for post order, pre order or in order traversals
* Implement push, pop operations of stack
* Implement enqueue and dequeue operations

Complexity analysis:

* Given a code, what is the order of complexity
* Given a code such as Fibonacci, prove the order of complexity
* Write the recurrence relation for quick sort or merge sort and prove the order of complexity

Programming:

* Know function objects (chapter-1)
* Know how stack can help in writing a code such as palindrome
* Know how to use find/union algorithms in a program