CS-151 **Final Exam** **May 16, 2012**

This Exam is being given under the guidelines of the **Honor Code**. You are expected to respect those guidelines and to report those who do not. Answer the questions in the spaces provided. If you run out of room for an answer, continue on the back of the page. There are 13 questions for a total of 132 points.

Name: _____

1. (5 points) Which of the following algorithms and/or data structures could be used to implement the Map ADT (That is, a data structure that acts like a Python dictionary)?

   - A List and the Binary Search algorithm.
   - A Heap
   - A Binary Search Tree
   - A Hash Table
   - A Stack

2. (2 points) Which of the above possibilities would be the best choice

3. (3 points) What are the three laws of recursion?

4. (10 points) Draw the expression tree for the expression $((8 * 7)/(4 + (3 * 5))) - ((16 * 3) + 4)$

5. (10 points) Recall that the height of the tree is defined as the number of edges between the root and the deepest leaf in the tree. Write a function height(t) that takes a tree as a parameter and returns the height of the tree. *Hint 1:* The methods for a tree are as follows:

   - BinaryTree() Create a new instance of a binary tree.
   - getLeftChild() Return the binary tree corresponding to the left child of the current node.
   - getRightChild() Return the binary tree corresponding to the right child of the current node.
   - setRootVal(x) Store the object in parameter x in the current node.
   - getRootVal() Return the object stored in the current node.
   - insertLeft(x) Create a new binary tree and install it as the left child of the current node.
   - insertRight(x) Create a new binary tree and install it as the right child of the current node.

   Hint 2: The height function is very simple (and short) if you think recursively

6. Given the following list of numbers x = [13, 24, 5, 7, 9, 17, 32, 27, 2]

   (a) (10 points) Create a binary search tree and insert each of the numbers. Draw a picture of the final tree. You do not need to show all the intermediate trees, just the final.

   (b) (10 points) Create a binary min-heap and insert the numbers one at a time into the heap. Draw both the tree and list representation of the heap after all the numbers are inserted.

7. (10 points) Given the following set of hash keys show the resulting hash table assuming that you are using linear probing. Insert them in order from left to right. The table holds exactly 11 keys. You do not need to worry about growing the table.

| 113 | 117 | 97 | 100 | 114 | 108 | 116 | 105 | 99 |

0	1	2	3	4	5	6	7	8	9	10

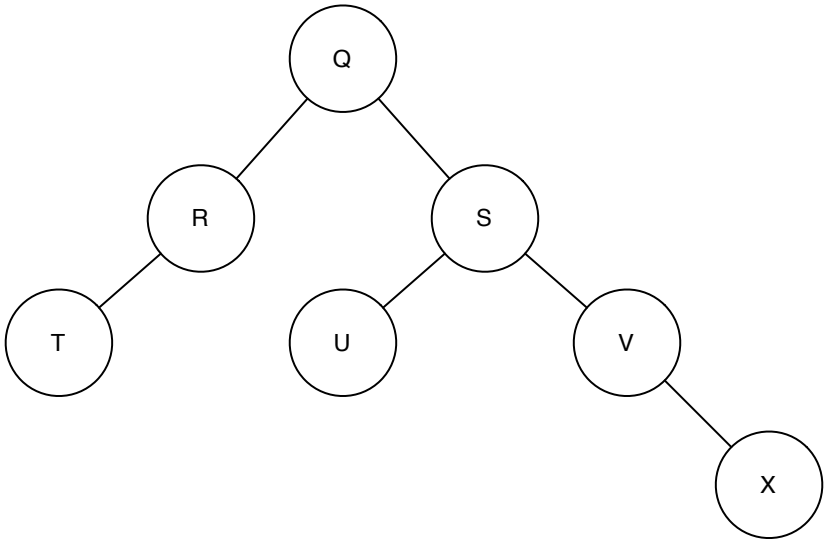
8. (10 points) Given the following list L = [90, 1, 3, 8, 56]. Show the contents of L after all passes of Insertion sort. I'll give you pass 1 for free!

90	1	3	8	56

9. (10 points) Given the following list L = [90, 1, 3, 8, 56, 21, 5, 13, 1, 146, 2, 34]. Show the contents of L after the After all the swapping is done for a gap size of 3 using Shell sort.

90	1	3	8	56	21	5	13	1	146	2	34

10. Given the binary tree shown below:



(a) (5 points) Perform a pre-order traversal of the tree. Write out the name of each node in the order it is visited.

(b) (5 points) Perform a post-order traversal of the tree. Write out the name of each node in the order it is visited.

11. Yertlenet is a primitive form of networking that was used at Luther college up until the mid 1980's. Yertlenet uses turtles as a message transport mechanism and requires that special one-way channels be dug between the buildings on campus to facilitate turtle navigation. A Message is routed from building x to building y by placing a message on the back of a turtle and setting the turtle in the channel leading to the next building. To facilitate routing, each building employed a student worker to receive incoming messages and if necessary move the turtle to the channel leading to the next building.

    You have just been hired as the new student worker for building one. Being a CS major you decide that you want to be the best turtle router on campus and therefore will create a graph of Yertlenet and figure out the optimum routing for messages that come through your building. The following table shows the information describing the Yertlenet connections. Column one is the start building, column two is the end building, and column three is the cost of the link.

```
1 2 10
1 3 15
1 6 5
2 3 2
3 4 7
3 6 10
4 5 7
6 4 5
5 6 13
```

    (a) (5 points) Draw the directed graph represented by the above table.

    (b) (12 points) Using Dijkstras algorithm, find the shortest path from each other building on campus to yours. Fill in the table below when you are done. You do not need to show every step of the algorithm, but make sure you show changes to the priority queue.

node	dist	pred
1		
2		
3		
4		
5		
6		

12. Consider the following classic problem: A farmer along with his fox, goose and a sack of oats are out for a walk. They come to a river that they must cross. The problem is that the boat is only large enough to hold the farmer and one additional object. The goose cannot be left alone with the grain or it will eat all the grain. The fox cannot be left alone with the goose or it will eat the goose. The goose, fox, and oats are incapable of rowing. Find a series of crossings that will get everyone safely to the other side of the river.

    This problem can be solved using a graph search algorithm. The key is to recognize that the nodes in the graph represent a "configuration" of the problem, with certain things on one side of the river and certain things on the other. Edges between nodes represent legal movements of items from one side to the other.

    (a) (10 points) Draw a graph to show a solution to this problem.

    (b) (5 points) Which Graph algorithm would you use to find the smallest number of crossings?

13. (10 points) Apply the depth first search algorithm to the graph below. Show the starting and finishing times for each vertex. Draw in the predecessor links for each vertex as well. Start with node **a**. You should assume that the vertices are stored in alphabetical order on the adjacency lists.