You should be prepared to answer questions about any of the material that I have covered in class or assigned in the readings. You should also be prepared to answer questions related to the programming assignments, or about Unix commands that you should have learned about in lab and while doing the assignments.

Review section? Ask Chris.

Topics to review (in no particular order) include:

- **Recursion**
  - read recursive method to determine results
  - write recursive method
  - convert simple recursion to iteration
- **Binary Search**
  - with BST
  - with sorted array
- **Binary Trees**
  - TreeNode
  - Methods: constructor, attachLeft/Right, attachLeft/RightSubtree, detachLeft/RightSubtree, getRootItem, setRootItem, isEmpty.
  - Traversals
    - preorder, inorder, postorder.
  - Recursive definition
  - Terminology: root, node, parent, child, sibling, full tree, balanced tree, etc.
  - Height
  - Advantages and disadvantages of trees when compared with linear data structures (lists, stacks, queues)
- **Binary Search Trees**
  - Methods: create, search, insert, retrieve, delete
  - Traversals
  - Definition
  - Typical and worst-case computational behavior of search, insert, delete, traversal
- **Java features**
  - Abstract class
  - Inheritance
  - Iterator – particularly constructor, hasNext(), next(), and methods for defining iteration order
- **Unix input/output redirection**
  - diff command
- **Basic C language details**
  - Understand a simple program
  - lint
  - Similarities/differences with Java
- **Plus everything from earlier this quarter**
Example questions

1) Given the following binary tree:

```
        a
       / \
      /   /
     b   c
    / \ / \
   d  e f  g
  /   \ /
 h    i
  \   /
   j
```

What node(s) is/are the parents of e?
What node(s) is are the child/ren of c?
Which nodes are leaf nodes?
What is the height of this tree?
Is this a balanced tree?

2) Draw a picture of the BST that is the result of inserting the following Strings into an initially empty tree: monkey, goose, duck, zebra, elephant, turkey, kangaroo, yak, cat.

3) What is the inorder traversal of the tree from question 1? postorder? preorder?

4) Write a recursive routine that prints a String in reverse order

5) Complete the following pseudocode to implement a preorder traversal of a binary tree. Display the contents of each node when you visit it.

   Traverse( BinaryTree bTree ) {

6) Assume you have a BST that contains Strings. What happens if you insert a sorted set of Strings into an initially empty BST?

7) What is the typical computational complexity (Big 'O') of the BST search algorithm?
What is the worstcase complexity?

8) What is the typical computational complexity of the BST postorder traversal algorithm? What is the worstcase complexity?

9) How are strings terminated in C?
10) Write a Java statement that would be the best translation of the following C statement (You may assume that n has been declared correctly).

```c
printf("n = %d\n", n);
```

What is the type of n in the above statement?

11) Which of the following values is the closest to the number of nodes that a binary search would examine in a BST with 1,000,000 nodes? Hint: $2^{10} = \text{approx } 103$.

   a. 1,000,000
   b. 20
   c. 1,000
   d. 20,000

12) Rewrite the recursive method mystery as an iterative method.

```java
public int mystery( int n ) {
   if ( n <= 0)
      return 0;
   else
      return n + mystery( n - 1 );
}
```
13) Write the binary search method that searches a sorted array of ints for a key value. Return the index of the array element if the array contains the key value. Return -1 if the array does not contain the key value.

   int search( int[] array, int key ) {

14) Use a TreeIterator to display the values stored in the nodes of a Binary Tree using a postorder traversal. The method FillTree() inserts String values into bTree.

   BinaryTree bTree = new BinaryTree();
   FillTree( bTree );

   // Your code goes here