Recitation Guide for February 25, 2008

I. Housing Keeping
   a. Homework 5 – Due Tuesday March 4th 12:01am midnight.
   b. Drop day – Friday February 29th.
   c. Pair-programmers' agreement – Due Wednesday March 5th in class (same day as Quiz 2)

II. Pair – Programming:
   a. General overview
      i. Driver – types in the code at the computer
      ii. Navigator – reviews the lines of code as it is typed in
      iii. More resources:
         1. http://coweb.cc.gatech.edu/cs1316/188
         2. http://en.wikipedia.org/wiki/Pair_programming
   b. Pair-programming assignments start with homework 6.
   c. Pairs must turn in a pair-programmer's agreement and post the pair to the coweb pairs page (http://coweb.cc.gatech.edu/cs1316/1035) to have the assignment graded.
      i. User: attach
      ii. Passwork: carmen

III. Problem Solving (optional)

IV. Recursion
   a. A recursive method is one that calls itself.
   b. "Divide and conquer" approach by dividing the problem into smaller and smaller sub-problems
   c. In mathematics, a recurrence relation is an equation that defines a sequence recursively. Sometimes it helps to find the recurrence relation before beginning to program.
   d. Example: Fibonacci numbers
      i. Defined by the recurrence relation: $F_n = F_{n-1} + F_{n-2}$ with $F_1 = 1$ and $F_2 = 1$
      ii. The sequence of Fibonacci numbers: 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89 ...
      iii. How would you approach this?
         1. Consider the base case(s)/ the smallest problem(s)
            a. The base cases are the problems that we can solve right away. For the Fibonacci numbers, the base cases are $F_1 = 1$ and $F_2 = 1$.
            b. We always need base cases or else we will never have a problem we can solve.
         2. Break the problem down into sub-problems
            a. The problem does not look like our base case(s), therefore we need to keep breaking it down until it is composed of the base

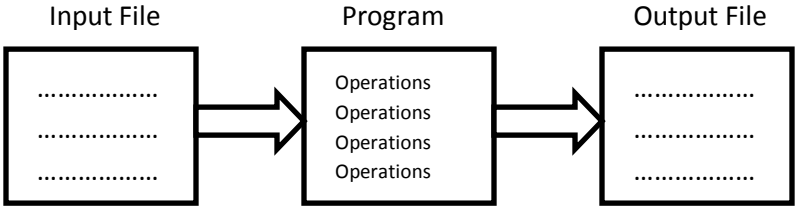
case(s). We break the problem down based on the relationship defined by the recurrence relation.

3. The code

```
public int fibonacci(int i){
    //base cases
    if (i == 1 || i == 2)
        return 1;
    //not a base case, break the problem down
    return fibonacci(i-1) + fibonacci(i-2);
}
```

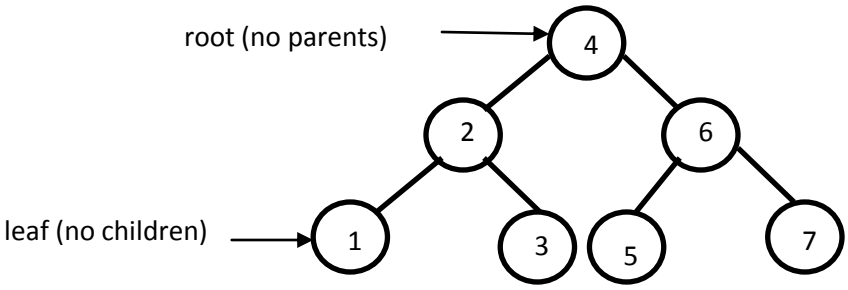
e. Example: Factorial
   i. Defined by the recurrence relation: $F_n = n * F_{n-1}$ with $F_0 = 1$
   ii. How would you approach this?
f. Example: Palidrome
   i. A **palindrome** is a word, phrase, number or other sequence of units that has the property of reading the same in either direction.
   ii. How would you approach this? (think more conceptually than code-wise)

V. File Input/Output (or File I/O pronounced "file-i -o")
   a. General idea: Retrieving data input from a file, performing some operations and outputting to another file.



   b. A delimiter is a sequence of one or more characters that separate the entries, or tokens, in the file. Common limiters: "\t" (tab), "\n" (enter/carriage return), "," (comma), " " (space)
   c. Code example: http://java.sun.com/docs/books/tutorial/essential/io/scanning.html

VI. Tree data structure
   a. A tree is acyclic (without cycles) graph.
      i. BST (Binary Search Tree)



ii. Each node has a max of two children.
iii. Usually sorted.
iv. Traversals
    1. Pre-order traversal (Parent, Left Child, Right Child)
        a. 4, 2, 1, 3, 6, 5, 7
    2. In-order traversal (Left Child, Parent, Right Child)
        a. 1, 2, 3, 4, 5, 6, 7
    3. Post-order traversal (Left Child, Right Child, Parent)
        a. 1, 3, 2, 5, 7, 6, 4
    4. Depth-first traversal
    5. Breadth-first traversal: http://en.wikipedia.org/wiki/Image:Animated_BFS.gif