**COMP 132 - Homework # 10**

**Recursion**

1. In class we traced the execution of several recursive function definitions. For example, we defined then traced as follows:

For each of the recursive function definitions given below, trace the execution of the indicated recursive function call. (Don’t worry about formatting your solution nicely. Any legible solution will not be penalized, and handwritten solutions are also acceptable.)

i. f(x) = 2\*x + f(x-1)

f(0) = 3;

Trace: f(4)

ii. g(x) = 2\*g(x-1) – g(x-2)

g(2) = 1

g(1) = 3

Trace: g(5)

2. Consider the following recursive method:

**public** **static** **void** func(**int** num) {

**if** (num == 5) {

System.*out*.println("done!");

}

**else** {

*func*(num+1);

System.*out*.println(num);

}

}

What output would be generated by the method call func(1)?

3. Each of the following recursive function definitions contains an error. Briefly but fully and clearly explain what is wrong with each definition.

i. f(x) = 3\*f(x-5) where x is a positive integer.

f(0) = 7

ii. g(x) = 2 - 4\*g(x) where x is a positive integer.

g(1) = 3

g(2) = 4

4. Give a recursive definition for each of the problems stated below. For each problem clearly state the base case(s) and the recursive case(s). You do not need to implement these definitions in Java.

i. exp(x,n) = xn , where both x and n are non-negative integers.

E.g. exp(2,5) = 32

ii. Assume we have a programming language that only allows conditionals (i.e. if statements), operations for +1 and -1, and recursive calls. How could the following function be defined recursively in that language?

add(a,b) = a + b both a and b are non-negative integers.

E.g. add(5, 7) = 12

5. In mathematics, the *factorial* function is defined for positive integers as follows: . Write a java method fact(int n) that computes (the factorial of n) recursively. Copy and paste the source code for your recursive method as your solution to this problem.