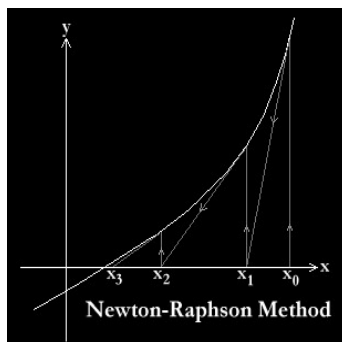


CAS CS 112 – Spring 2008, Assignment 1

Problems 1-3 due at 10:00 pm on Thursday, January 31

Problem 4 (written) due at the beginning of class on January 31

Problem 1, Approximating Roots



(30 points) The Newton-Raphson method can be used to approximate the roots of a function (the roots of $f(x)$ are the values x where $f(x)$ is zero). We start by taking a guess x_0 , and then improving it by considering the point where the tangent to $f(x)$ at x_0 intersects the x-axis, let us call this point x_1 (see diagram above). x_1 will be closer to the root, and we can find x_2 by finding the intersection with the x-axis of the tangent to the curve at x_1 , and so on. It is easy to see that given x_n ,

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}.$$

How is this useful for finding roots of real numbers? Consider that finding the k th root of a non-negative real number n is equivalent to finding the root of $f(x) = x^k - n$. Your job is to write a recursive function, which given k , n , x_0 and $iter$, computes an approximation of the k th root of n by performing $iter$ steps of Newton-Raphson starting at x_0 . Submit a single file called **NewtonRaphson.java** which has a main containing a loop asking the user for arguments, and calls a recursive function **float approximate (int k, float n, float x0, int iter)** that returns the approximation. The program should terminate when a negative k is input. You can read more about the Newton-Raphson method here: http://en.wikipedia.org/wiki/Newton's_method.

Problem 2, Heffalumps and Woozles

(25 points) As is well known, heffalumps and woozles multiply very quickly, without any need for mating (they appear in a Disney movie, after all). A heffalump produces three heffalumps and two woozles in its lifetime. A woozle produces one heffalump and five woozles in its lifetime. Your program should repeatedly read an integer n from the console and print out the total number of heffalumps and woozles in n generations, assuming the first generation has one heffalump and one woozle. The program should terminate when a negative n is input. Submit a single file called **Heffalumps.java** which has a main

to read in n , and a function `int count (int numHeffalumps, int numWoozles, int generations)`.

Problem 3, Wally Gone Wild



(25 points) Remarkably, we now have another organism that reproduces asexually, let us call him Wally (like the Red Sox mascot). Each generation a mature Wally gives birth to another Wally. It takes Wally one generation to reach maturity, and a Wally never dies. If in generation 1, we have 1 newborn Wally, how many Wallies will there be in generation n ? Please formulate a recursive and iterative solution to the problem. What is the complexity of each one (in terms of n)? Submit a single file called **Wally.java** with two methods: **countWallyRecursive** and **countWallyIterative**. Your main should be in the same file, it should repeatedly read an integer n from the console and print out the total number of Wallies in generation n twice (once using each function). The main loop should stop when a negative n is entered. State the complexity of each function in the comments. You can read about Wally and his rivalry with Raymond, the Tampa Bay mascot, here: http://en.wikipedia.org/wiki/Wally_the_Green_Monster.

Problem 4, Asymptotic Notation

(20 points) Order the following functions in order of growth from smallest to largest, so that each function is O of the next one. State which functions are Θ of each other.

$6n \log n$, 2^{1000} , $\log \log n$, $(\log n)^2$, $2^{\log n}$, 2^{2^n} , \sqrt{n} , $n^{0.01}$, $\frac{1}{n}$, $4n^{1.5}$, $5n$, $3n^{0.5}$, n^3 , $4^{\log n}$, 4^n , $2n \log n^2$, $n \ln 4n$, $\sqrt{\log n}$, $n^2 \log n$, 2^n .

Please submit the answer to this question at the start of class on the due date.