Recursion

-Brian McClintock

Ryan Reid

Baibhab Dutta

Recursion is the process of defining something in terms of itself. In java, recursion is the attribute that allows a method to call itself. Thus, in the course of the function definition there is a call to the same function. A method that calls itself is said to be recursive.

Infinite Recursion

In simple words, infinite recursion behaves like a never ending loop where the function is called over and over again. This is a common programming error. The computer needs to keep some amount of memory for bookkeeping for each call. After a few method calls this memory will be exhausted and the program will shut down reporting a "stack overflow" error. Infinite recursion happens either because the parameter values don't get simpler or because a special terminating case is missing.

How recursion works?

Just like iteration, the process of recursion also depends on a certain condition. In practice we check to see if that certain condition is true and in that case exit (return from) our method. *The method in which we end our recursion is called the base case.* A recursive method behaves just like a loop, where there should be a control variable whose value is subject to change, thereby advancing closer to the base case, each time the method calls itself.

Types of Recursion

Linear Recursion: A linear recursive function is a function that makes only a single call to itself each time the function runs.

Binary Recursion: A recursive function which calls itself twice during the course of its execution.

Exponential Recursion: Recursion where more than one call is made to the function from within itself. This leads to exponential growth in the number of recursive calls.

Nested Recursion: In nested recursion, one of the arguments to the recursive function is the recursive function itself. These functions tend to grow extremely fast.

Tail Recursion: A recursive procedure where the recursive call is the last action to be taken by the function. Tail recursive functions are generally easy to transform into iterative functions.

Head Recursion: A call is head-recursive when the first statement of the function is the recursive call.

Middle or Multi Recursion: A call is mid-recursive when the recursive call occurs in the middle of the function. I.e. there are other statements before and after the recursive call. If one or more of these statements is another recursive call, then the function is multi-recursive. There is no essential difference between Head Recursion, Middle Recursion and Multi Recursion from the standpoint of efficiency and algorithm theory.

Mutual Recursion: Function X and Y are called mutually-recursive if function X calls function Y and function Y in turn calls function X. This is also called indirect recursion because the recursion occurs in two steps instead of directly.