Types of Recursion

Recursive functions can be classified on the basis of :  
a.) If the functions call itself directly or indirectly –**Direct / Indirect**  
b.) If an operation is pending at each recursive call – **Tail Recursive/ Not**  
c.) based on the structure of the function calling pattern – **Linear / Tree**

Direct Recursion:

* If a function explicitly calls itself it is called directly recursive.
* When the method invokes itself it is direct.
* int testfunc(int num) {
* if (num == 0)
* return 0;
* else
* return (testfunc(num - 1));
* }

Here, the function ‘testfunc’ calls itself for all positive values of num.

## Indirect Recursion

* This occurs when the function invokes other method which again causes the original function to be called again.
* If a method ‘X’ , calls method ‘Y’, which calls method ‘Z’ which again leads to ‘X’ being invoked is called indirect recursive or mutually recursive as well.

int testfunc1(int num) {

if (num == 0)

return 0;

else

return (testfunc2(num - 1));

}

int testfunc2(int num2) {

return testfunc1(num2 - 1);

}

## Tail / Bottom Recursion

A function is said to be tail-recursive, if no operations are pending when the recursive function returns to its caller.

* Such functions, immediately return the return value from the calling function.
* It is an efficient method as compared to others, as the stack space required is less and even compute overhead will get reduced.

int fact(int n){

if (n == 1)

return 1;

else

return (n \* fact(n - 1));

}

In order to make it tail recursive, information about pending tasks has to be tracked.

int fact(int n){

return (n \* fact2(n - 1));

}

int fact2(int n, int result) {

if (n == 1) return result;

return fact2(n - 1, n \* result);

}

**Linear Recursion:**

 It is the most popular recursion method. When using this method, functions call themselves in a non-complex manner and terminate the function with a termination condition. This method involves functions making a single call to itself during execution.