General form of Recursion

<ReturnType> name(Parameter){

//base condition

// recursive call

}

A calling phase and a returning phase.

Void test(){

If(base){

Calling phase

Test();

Returning phase

}

Type of Recursion:

Tail recursion

Head recursion

Tree Reursion

Indirect recursion

Nested recursion

TAIL RECURSION:

Tail recursions can easily be converted to a loop , thus many compiler converts the tail recursion function to a loop to save / optimize the additional overhead of memory.

Tail recursion are the recursive function which calls itself only at the very end of the function. There is no additional call left after the function.

Void test(){

//some code

test();

}

HEAD recursion:

Head recursions are the recursive function which calls itself at a point when there are still more operation pending in the stack.

You can’t convert the head recursion to loop.

For example print 1,2,3..n

Void headRec(int n) {

If(n>0){

headRec(n-1);

print(n);

}

But the loop can ‘t be easily done. We would have to do more work .

TREE Recursion:

If the recursive function calls itself more one time in the same stack frame, its called Tree Recursion.

O(2^n) and O(h) space complexity

INDIRECT RECURSION:

Call difrent recursions of two or more recursion

NESTED RECUSION

Int fun(int n){

If(n>100){

Return n-10;

}else

return Fun(fun(n-1));

}

}