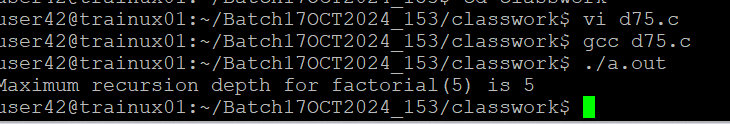
# Recursive Function Assignment

1. WAP to calculate the maximum stack depth of a recursive call to a function. (For eg a factorial function ).

A screenshot of a computer

Description automatically generated



1. What is tail recursion? Why is it important? Give an example

Tail recursion is a special type of recursion where the recursive call is the last operation in the function before it returns a result. In other words, when a function is tail-recursive, it performs its recursive call as the very last action and returns the result directly without needing to perform any further work after the call.

Tail recursion is important because it can be optimized by the compiler or interpreter through **tail call optimization (TCO)**. Tail call optimization allows the compiler to reuse the current function’s stack frame for the next recursive call, thereby **avoiding stack overflow** and making recursive calls as efficient as an iterative loop.

Without tail recursion, each recursive call consumes a new stack frame, which increases memory usage. However, with tail recursion, the recursive calls are made without adding new stack frames, as the current function call is completed and its stack frame can be reused for the next call.

In some programming languages (like Scheme, Lisp, or even some implementations of C), tail recursion can be automatically optimized into a loop, making recursive functions much more memory-efficient.

**Tail Recursion in the factorial\_tail\_recursive function:**

* The function factorial\_tail\_recursive takes two arguments: n (the number for which we want to calculate the factorial) and accumulator (which accumulates the result as the recursion progresses).
* The base case checks if n == 0 or n == 1, in which case the accumulator value is returned.
* In the recursive call, the accumulator is updated by multiplying it by n, and this updated accumulator is passed into the next recursive call. This is the key to tail recursion: the recursive call is the last operation and directly returns the result.