**ASSIGNMENT-3:** Function Design and Modularization - Create a document that describes the design of two modular functions: one that returns the factorial of a number, and another that calculates the nth Fibonacci number. Include pseudocode and a brief explanation of how modularity in programming helps with code reuse and organization.

**SOLUTION:**

A function that returns the factorial of a number:

**DESCRIPTION:** The factorial function calculates the factorial of an positive integer n. It is denoted by n! The factorial of a nth number is product of all positive numbers less than or equal to the integer n.

fact(n):

If n==0 or n==1

Return 1

else

Return n\*fact(n-1)

A Function that calculates the nth Fibonacci number:

**DESCRIPTION:** The fibonacci function calculates the nth fibonacci number. The fibonacci sequence is a series of numbers in which each number is the sum of preceding two numbers, usually this series starts with the number 0 and 1.by using calling same function repeatedly by its self we can get the result, this type of functions are known are recursive functions.

Fibonacci(n):

if n <= 1

return n

else

return Fibonacci(n - 1) + Fibonacci(n - 2)

\*Modularity in programming is a design principle involves breaking down a program into smaller, independent modules or functions, each module or function carries a specific task.

\*Code Reuse: Modularity allows develops to create reusable modules or functions that can be easily utilized in different parts of the program. Instead of duplicating code to perform similar tasks, we can simply call that specific modular function where needed.

\*Organization: By breaking down complex tasks into smaller, more manageable units, it improves code readability and makes it easier for developers to understand and navigate. This organization not only facilitates collaboration among developers but also simplifies onboarding for new team members who may need to understand and work with the code.

\*By summarizing we can say that modularity in programming significantly contributes to code reuse and organization by promoting the creation of reusable, maintainable, and well-structured modules or functions.