Designing Modular Functions for Factorial and Fibonacci Calculation

Modularity in programming is the practice of breaking down a program into smaller. This approach promotes code reuse and organization, making the code more manageable and easier to understand.

Pseudocode for Factorial of number

Int factorial(int n) {

If (n < 0) {

Return -1; // Error: Factorial not defined for negative numbers

} else if (n == 0) {

Return 1;

} else {

Int result = 1;

For (int I = 1; I <= n; i++) {

Result \*= I;

}

Return result;

}

}

Pseudocode for Nth Fibonacci number

If n < 0 then

Return

Elseif n <= 1 then

Return n

Else

Return fibonacci(n-1) + fibonacci(n-2)

Endif

End function

Advantages of Modularity :

\* Code Reusability: Modular functions can be reused in different parts of the program or even in other programs, reducing code duplication.

\* Improved Organization: Dividing code into functions enhances readability and maintainability by separating functionalities into logical units.

\* Easier Testing: Well-defined functions can be independently tested, simplifying debugging.