**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 in programming helps with code reuse and organization.**

**ALGORITHM:**

**FACTORIAL**

1.Start

2.take input function factorial(n)

3.if n is 0 return 1

4.else

5.return n \* factorial(n - 1)

6.end

**Explanation**: If the input is 0 it returns 1 otherwise it multiplies the number with the factorial of the previous number.

**FIBONACCI**

1.start

2.Take input function fibonacci (n)

3.if n is 0 return 0

4.else

5.if n is 1 return 1

6.else

6.return fibonacci(n - 1) + fibonacci(n - 2)

**Explanation**:. If the input is 0 it returns 0 if it's 1 it returns 1. For any other input it returns the sum of the Fibonacci numbers at positions (n - 1) and (n - 2).

**PSEUDOCODE :**

**FACTORIAL**

function factorial(n)

if n == 0

return 1

else

return n \* factorial(n-1)

**FIBONACCI**

function fibonacci(n)

if n <= 1

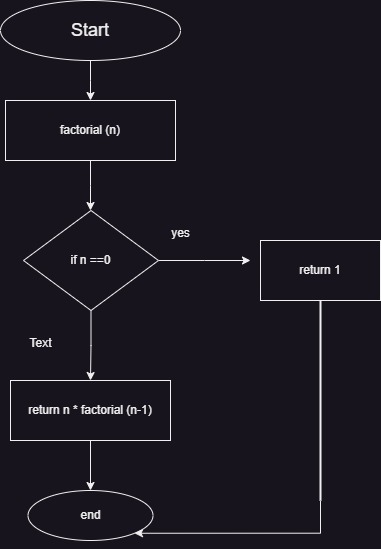
return n

else

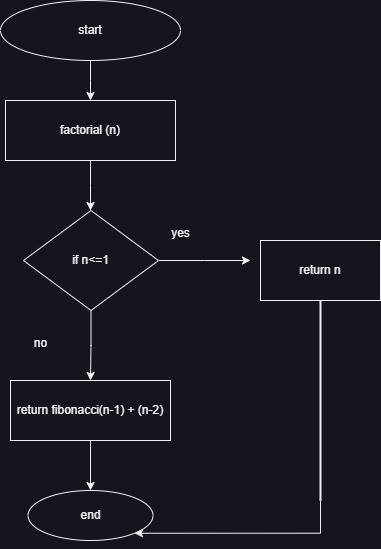
return fibonacci(n-1) + fibonacci(n-2)

**FLOWCHART :**

**1.FACTORIAL**



**2.FIBONACCI**



**EXPLAINATION:**

**Code reuse :**Modular functions reused in all different parts of program mainly its saves time

**Organization:** it makes easier to understand and debug