**Recursive Function and Efficiency Analysis - Write a recursive function pseudocode and  
calculate the nth Fibonacci number and use Big O notation to analyze its efficiency. Compare this with an iterative approach and discuss the pros and cons in terms of space and time complexity**.

**PSEUDOCODE**

**RECURSIVE FIBONACCI FUNCTION**

Fibonacci function of (n)

if n <= 1

return n

else

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

**ITERATIVE FIBONACCI FUNCTION**

iterative fibonacci function of(n)

if n <= 1

return n

else

starting fib\_value = 0

current fib\_value = 1

for i from 2 to n

next fib\_value = starting fib\_value + current fib\_value

previous fib\_value = current fib\_value

current fib\_value = next fib\_value

return current fib\_value

**PROS AND CONS OF SPACE AND TIME COMPLEXITY** :

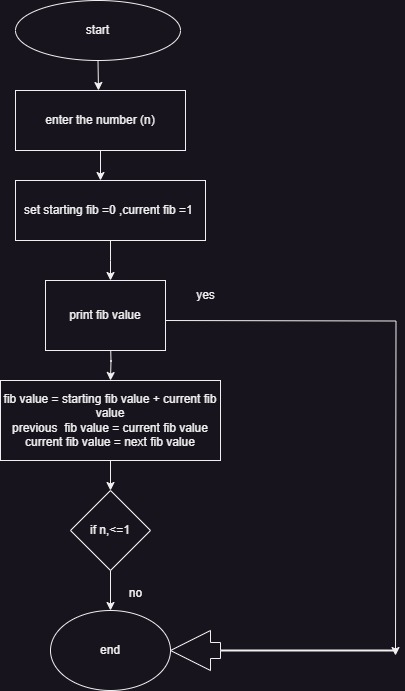
**Pros:**

* More efficient time complexity for large values of n.
* Requires less memory as it does not rely on additional space on the call stack.

**Cons:**

* Less intuitive implementation compared to the recursive approach.
* Overall the iterative approach is preferred in practice due to its superior time and space efficiency especially for large values of n. It provides a more scalable solution for calculating Fibonacci numbers with better performance and reduced memory usage.

**FLOWCHART**

****