Assignment 1: Pseudocode Development - Task: Write a detailed pseudocode for a simple program that takes a number as input, calculates the square if it's even or the cube if it's odd, and then outputs the result. Incorporate conditional and looping constructs.

START

REPEAT

DISPLAY "Enter a number:"

READ number

IF number MOD 2 = 0 THEN

result ← number \* number

DISPLAY "The number is even. Square =", result

ELSE

result ← number \* number \* number

DISPLAY "The number is odd. Cube =", result

END IF

DISPLAY "Do you want to continue? (yes/no):"

READ choice

UNTIL choice = "no"

END

Assignment 2: Flowchart Creation - Design a flowchart that outlines the logic for a user login process. It should include conditional paths for successful and unsuccessful login attempts, and a loop that allows a user three attempts before locking the account.

<https://lucid.app/lucidchart/75d13599-f596-4a90-a273-9fb6301c0e7b/edit?view_items=X2dAnb4hvfoe&invitationId=inv_53037f7a-71ec-45ed-aabf-5fc072ed34a9>

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.

1.Factorial

Calculates the factorial of a number using Iteration

PESUDO CODE:

Function Factorial (n: Integer) Returns Integer

If n == 0 Then

Return 1

End If

result ← 1

For i from 1 to n

result ← result \* i

End For

Return result

End Function

2.Fibonacci

Returns the nth Fibonacci number using Iteration

PESUDO CODE:

Function Fibonacci(n: Integer) Returns Integer

If n == 0 Then Return 0

If n == 1 Then Return 1

prev ← 0

curr ← 1

For i from 2 to n

temp ← curr

curr ← curr + prev

prev ← temp

End For

Return curr

End Function

IMPORTANCE OF MODULARITY:

It breaks a program into smaller, manageable, reusable pieces

Reusability: Functions can be reused in other programs

Maintainability: Bugs are easy to isolate and fix in small units

Readability: Easier to understand and Debug

Testing: Can be test Individually