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.

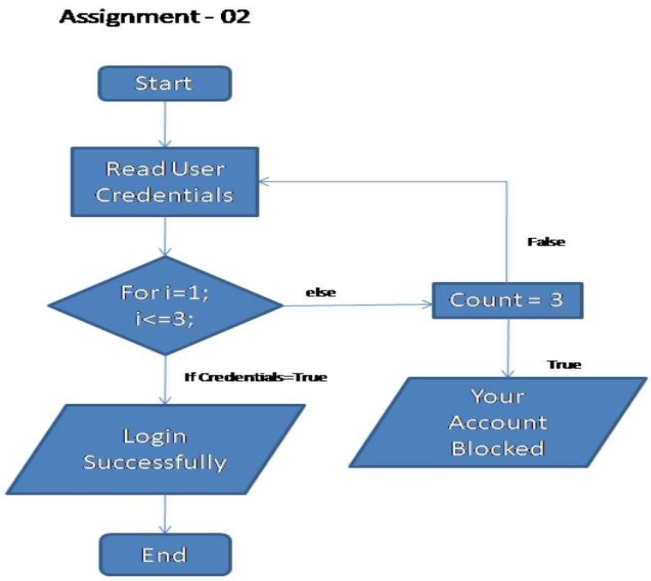
Solution:-

1. Start
2. Declare variables: num, result
3. Input num from the user
4. If num is even:
   a. Calculate result as num * num (square)
5. Else:
   a. Calculate result as num * num * num (cube)
6. Output result
7. 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.

Solution:-



Assignment 3

Create a Document that describes the design of 2 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:-

FACTORIAL OF A NUMBER:

```
Function Factorial(n):
    If n is less than or equal to 1:
        Return 1;
    Else:
        Return n * Factorial(n - 1);
```

Explanation: The factorial of a number n is the product of all positive integers less than or equal to n. This function uses recursion, where the function calls itself with a decremented value of n until it reaches the base case of n <= 1.

FIBONACCI OF A NUMBER:

```
Function Fibonacci(n):
    If n is 0:
        Return 0;
    Else if n is 1:
        Return 1;
    Else:
        Return Fibonacci(n - 1) + Fibonacci(n - 2);
```

Explanation: The Fibonacci sequence is a series of numbers where each number is the sum of the two preceding ones, usually starting with 0 and 1. This function uses recursion to calculate the nth Fibonacci number by summing the two preceding Fibonacci numbers until it reaches the base cases of n=0, n=1.

Modularity in Programming:

Modular programming is defined as a software design technique that focuses on separating the program functionality into independent, interchangeable methods/modules. Each of them contains everything needed to execute only one aspect of functionality.

Talking of modularity in terms of files and repositories, modularity can be on different levels -

- Libraries in projects
- Function in the files
- Files in the libraries or repositories

Modularity is all about making blocks, and each block is made with the help of other blocks. Every block in itself is solid and testable and can be stacked together to create an entire application. Therefore, thinking about the concept of modularity is also like building the whole architecture of the application.

Examples of modular programming languages - All the object-oriented programming languages like C++, Java, etc., are modular programming languages.