Day1-Exercise-Assignment-Shintu Kumar

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

print " Enter a number "

input number

if ( number % 2 == 0 )

print "Even" +(number\*number)

else

print "Odd"+ (number\*number\*number)

end loop

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.

Machine generated alternative text:
False 

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.

Function Design and Modularization

1. Factorial Function

factorial(num)

Input integer num

Output integer result

Pseudocode -

-> function factorial(num):

if num == 0

return 1

else

result = 1

for i from 1 to n:

result = result \* i

return result

The integer number is passed in function factorial. If number is 0 then return 1 and if number is greater the it will go in else condition where result declared as 1 and for loop running i from 1 to number and multiplying i in result and returrning the result.

2.Fibonacci Function:

fibonacci(n)

Input: integer n

Output: integer result

Pseudocode-

-> function fibonacci(n):

if n==0

return 0

else if n==1

return 1

else:

prev1 = 0

prev2 = 1

for i from 2 to n:

result = prev1 + prev2

prev1 = prev2

prev2 = result

return result

This function calculates the nth Fibonacci number. If n is 0, it returns 0; if n is 1, it returns 1. For n greater than 1, it uses a loop to calculate each Fibonacci number iteratively from 2 to n by summing the previous two Fibonacci numbers and then putting prev1 equal to prev2 and prev2 equals result to maintain series.

Modularity - Practice of breaking down a program into smaller, manageable, and reusable components called modules.These modules have specific functionalities, making the code easier to understand and maintain.

Advantages -

Modularity helps in code reusability, enhances organization, simplifies debugging and testing making code easy to maintain.