**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.

-Take input value from a user

-Read input number.

-Then check a user entered value is even number or odd number.

-check input number is even

(checking even number - input\_number%2 == 0)

if number is even

-calculate the square of the input number.

square = input\_ number \* input\_ number

-output -> square number.

------

or

------

else

number is odd

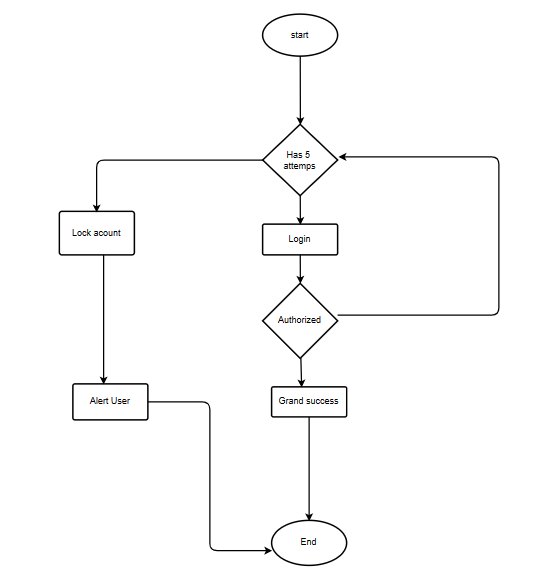
-calculate the cube of the input number

cube = input number \* input number \* input number

-output -> cube

**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.



**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."

FACTORIAL OF A NUMBER:

function: factorial(n)

input: An integer n

output: The factorial of n

Pseudo code:

step1: start

step2: Initialize a variable called result to 1.

step3: if n is 0 or 1 , return 1

Step4: Otherwise, for i from 2 to n , the following:

-Multiply result by i

step5: Return the value of result.

step6: Stop

**Assignment 4:**

Pseudocode and Flowchart for Sorting Algorithm - Write pseudocode and create a flowchart for a bubble sort algorithm. Provide a brief explanation of how the algorithm works and a simple array of integers to demonstrate a dry run of your algorithm.

Step 1: Start

Step 2: Read the array of given items from the user.

Step 3: Take the first element (index = 0), compare the current element with the next element.

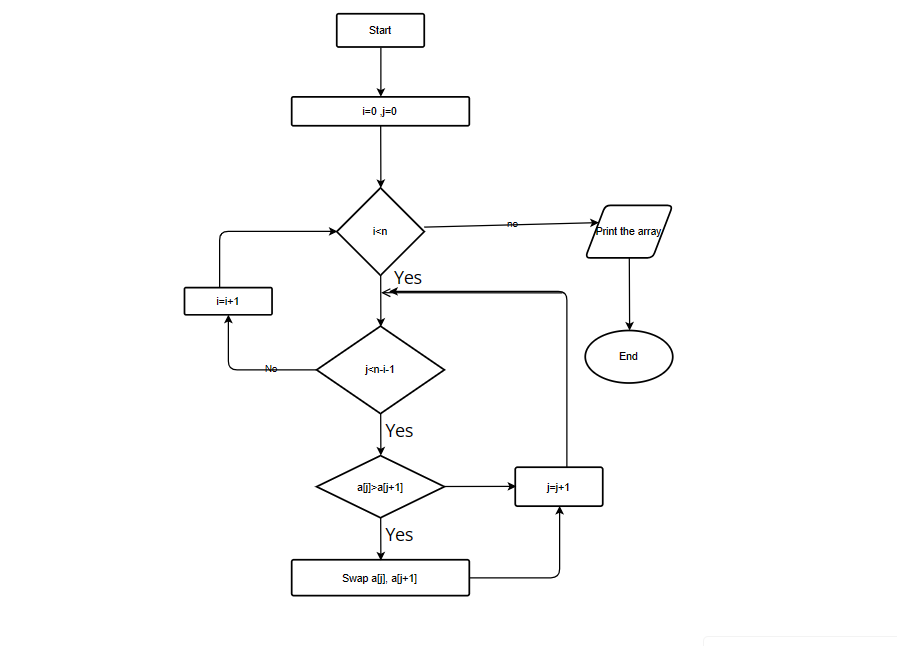
Step 4: If the current element is greater than the next element, swap them.

Step 5: Else,

If the current element is less than the next element, then move to the next element.

Step 6: Repeat Step 3 to Step 5 until all elements are sorted.

Step 7: Stop



**Assignment 5:** 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."

int recursiveFun1(int n)

{

if (n <= 0)

return 1;

else

return 1 + recursiveFun1(n-1);

}

int recursiveFun2(int n)

{

if (n <= 0)

return 1;

else

return 1 + recursiveFun2(n-5);

}

int recursiveFun3(int n)

{

if (n <= 0)

return 1;

else

return 1 + recursiveFun3(n/5);

}

void recursiveFun4(int n, int m, int o)

{

if (n <= 0)

{

Print ("%d, %d\n", m, o);

}

else

{

recursiveFun4(n-1, m+1, o);

recursiveFun4(n-1, m, o+1);

}

}

int recursiveFun5(int n)

{

for (i = 0; i < n; i += 2) {

}

if (n <= 0)

return 1;

else

return 1 + recursiveFun5(n-5);

}