Day2\_Exercise\_Assignment\_Shintu Kumar

Assignment 1: 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.

**Flowchart:**

Machine generated alternative text:


**Pseudocode:**

Bubble sort

1. Start

2. Declare an array named arr with size n

3. Run a for loop to take n inputs from user

4. iterate over array to sort

        for(int i=0; i<n; i++)

for(int j=0; j<n-i-1; j++)

if(arr[j] > arr[j+1])

arr = swap(arr,arr[j],arr[j+1])

end of inner for loop

5.         end of outer for loop

6. print sorted array temp

7. end

1. function swap(arr,num1, num2)

2. declare variable temp

3. int temp = num1

        num1 = num2

        num2 = arr

4. return arr

6. end

Bubble sort works by repeatedly traversing over an array by comparingadjacent elements and this puts the largest element at last position then again it traverse from 0 while now it goes till 2nd last element as last element was fixed in 1st iteration and goes further.

arr = [10,5,6,33,1]

i        j        j+1        arr[0]        arr[1]        arr[2]        arr[3]        arr[4]

i=0        j=0        1        5        10        6        33        1

i=0        j=1        2        5        6        10        33        1

i=0        j=2        3        5        6        10        33        1

i=0        j=3        4        5        6        10        1        33

i=1        j=0        1        5        6        10        1        33

i=1        j=1        2        5        6        10        1        33

i=1        j=2        3        5        6        1        10        33

i=2        j=0        1        5        6        1        10        33

i=2        j=1        2        5        1        6        10        33

i=3        j=0        1        1        5        6        10        33

Assignment 2: Recursive Function and Efficiency Analysis - Write a recursive function pseudocode and calculate the nth Fibonacci number and use Big O notation to analyse its efficiency. Compare this with an iterative approach and discuss the pros and cons in terms of space and time complexity.

function fibonacci recurssion

1. start

2. take variable num till which series will be shown

3. make function fibo and pass num in it with number recieving ans

4. print(number+" ")

5. end

---------------------------------------------------------

function fibo(num)

if(num ==1)

return 0

else if(num == 2)

return 1

else

fibo(num-1)+fibo(num-2)

if end

stop

time complexity = O(2^n)

---------------------------------------------------------

function fibonacci\_iterative(n)

array fib[n]

if n <= 1

return n

else

fib[0] = 0

fib[1] = 1

for i from 2 to n

fib[i] = fib[i-1] + fib[i-2]

return fib[n]

stop

time complexity = o(n)

The recursive approach has exponential time complexity O(2^n) but linear space complexity O(n), while the iterative approach has linear time complexity O(n) and linear space complexity O(n). The recursive approach is less efficient due to the exponential growth of recursive calls, especially for large values of n, while the iterative approach is more efficient and suitable for practical use, particularly for large inputs.