RECURSION : Guided Exercises

1. Write the recursive algorithm that calculates the sum of the first n natural numbers.
2. Write recursive algorithms for:
   (a) the quotient of a divided by b,
   (b) the remainder of a divided by b,
   (c) the greatest common divisor (GCD) of two non-negative integers.
3. Provide a recursive definition of a + b, where a and b are non-negative integers. Write the corresponding recursive algorithm.
4. Given an array A of integers, present recursive algorithms for:
   (a) finding the maximum value in A,
   (b) finding the minimum value in A,
   (c) calculating the sum of the elements in A,
   (d) calculating the product of the elements in A,
   (e) finding the average of the elements in A.
5. Develop both iterative and recursive algorithms for:
   (a) calculating the factorial,
   (b) calculating the product a * b,
   (c) generating the Fibonacci sequence.

Evaluate (trace) the algorithms for the following cases: 6!; 9!; 100 * 3; 6 * 4; fib(10); fib(11). Compare the iterative and recursive algorithms.

6. Recursive algorithms on linked lists (LL):
   (a) reversing an LL,
   (b) searching for a given element in an LL.
7. Given the following PASCAL recursive definition:

```
FUNCTION Fib(N: INTEGER): INTEGER;
VAR X, Y: INTEGER;
BEGIN
  IF N <= 1 THEN
    Fib := N
  ELSE
    X := Fib(N-1);
    Y := Fib(N-2);
    Fib := X + Y;
  ENDIF;
END;
```

(a) Provide a fully parenthesized expression for fib(6), and then evaluate it.
(b) Define the points of return for the procedure.
(c) What does the data area contain? Define its structure.
(d) Transform the recursive algorithm using a stack. The resulting algorithm is informal.
(e) Provide a trace.
(f) Can you reduce the data area? Redefine the data area accordingly.

(g) Attempt to make the algorithm formal. This involves removing the "GOTOs." Can you completely eliminate the stack?

8. Repeat the same process for the following PASCAL function:

```
FUNCTION Comb(N, M: INTEGER): INTEGER;
{ Calculate C(N, M) for 0 ≤ M ≤ N and N ≥ 1 }
BEGIN
  IF (N = 1) OR (M = 0) OR (M = N) THEN
    Comb := 1
  ELSE
    Comb := Comb(N-1, M) + Comb(N-1, M-1);
  ENDIF;
END;
```

9. Transform the recursive algorithm corresponding to the Tower of Hanoi.

10. Revisit the recursive algorithm that reverses a singly linked list from exercise 6, and then use a stack to eliminate the recursion.