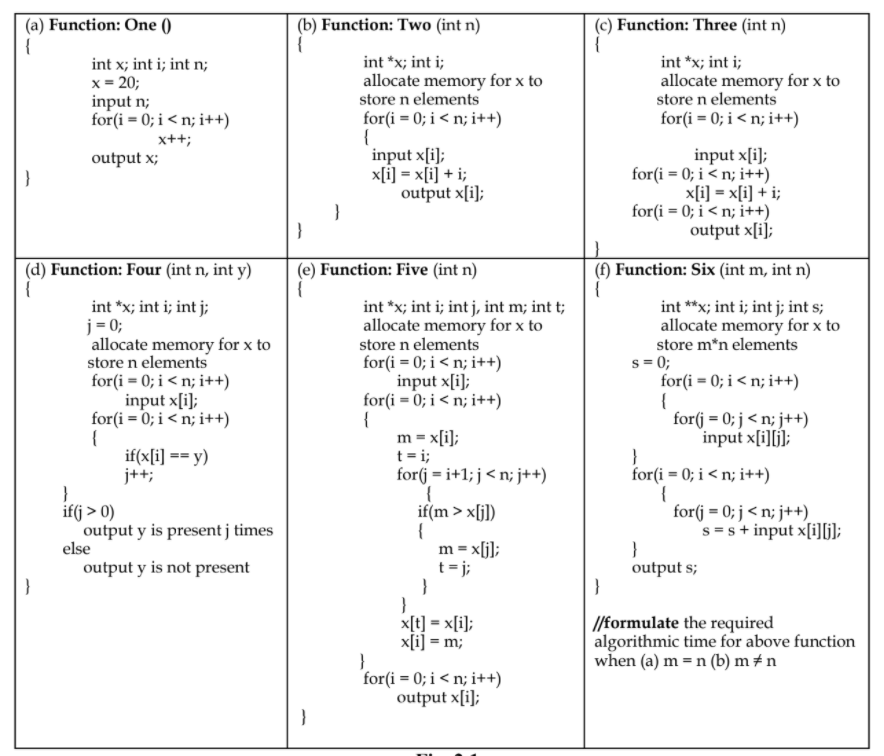
**Algorithms and Problem-Solving Lab (15B17CI471)**

**EVEN 2022**

**Week -2 (14 Feb - 19 Feb 2022)**

1. You are given an array A[m] where first n cells contain integers in sorted order and the rest of the cells are filled with 0. Here assumes m>>n and value of n is unknown. Implement an algorithm that takes an integer x as input and finds a position in the array containing x, if such a position exists, in O (log n) time.
2. Assume that we are given *n* pairs of items as input, where the first item is a number and the second item is one of three colours (red, blue, or yellow). Further assume that the items are sorted by number. Give an *O*(*n*) algorithm to sort the items by colour (all reds before all blues before all yellows) such that the numbers for identical colours stay sorted. For example: (1, blue), (3, red), (4, blue), (6, yellow), (9, red) should become (3, red), (9, red), (1, blue), (4, blue), (6, yellow).
3. Find the complexity of the following code snippets:



1. Implement the recursive algorithms for (a) Tower of Hanoi and (b) Fibonacci Number computation and analyse the space and time requirements of both the algorithms.
2. Implement the algorithm (Algo\_1) presented below and discuss which task this algorithm performs. Also, analyse the time complexity and space complexity of the given algorithm. Further, implement the algorithm with following modification: replace m = ⌈2n/3⌉ with m = ⌊2n/3⌋, and compare the tasks performed by the given algorithm and modified algorithm.

Algo\_1(A [0 ... n-1])

{

if n = 2 and A[0] > A[1]

swap A[0] ↔ A[1]

else if n > 2

m = ⌈2n/3⌉

Algo\_1 (A [0 .. m − 1])

Algo\_1 (A [n – m .. n − 1])

Algo\_1 (A [0 .. m − 1])

}