Unique Paper Code	: 42347902
Name of the Course	: B.Sc. Programme / B.Sc. Mathematical Science
Name of the Paper	: Analysis of Algorithms and Data Structures
Semester	: V
Year of Admission	: Upto 2018

Duration: 3 Hours Maximum Marks: 75

Instructions for candidates:

1. All questions carry equal marks.
2. Attempt any **FOUR** Questions.

1. Consider the pseudocode given below to sort a set of numbers in increasing order:

```
my_sort(lst)
    for i = 0 to length(lst) - 2
        k = i
        for j = i to length(lst)-1
            if lst[j] < lst[k]
                k = j
        swap(lst[i], lst[k])
    print(lst)
```

Specify the best and worst case runtime complexity of my_sort(). Perform my_sort() on the input list [3, 2, 4, 6, 0, 7, 5, 1] and show the results obtained after every iteration.

If you have insertion sort also to work with, which one of the two (insertion/my_sort) will you prefer for an input list which is almost sorted? Justify with reasons.

Consider a scenario where insertion sort and merge sort are available to sort an array of integers in ascending order. For each algorithm, determine the time complexity if the input is described as: Case 1 - already sorted, Case 2 - sorted in descending order, and Case 3 - contains all elements with equal values. Justify with reasons.

2. Construct any possible Binary Search Tree for the following values and perform the inorder, postorder, preorder, and level order traversals.

8, 5, 1, 7, 10, 12, 14, 3, 6

Next, consider the pseudocode below:

```
FindMe(node)
    if (node != Null)
        FindMe(node.right)
        if (node.key % 2 != 0)
            print(node.key)
        FindMe(node.left)
```

For the tree constructed, what is the output of the above piece of code? In what ways does the result relate to any of the above-mentioned traversals?
Starting form the original code, find the output for the following scenarios and write how the result relates to any of the above-mentioned traversals: Case 1: What will be the output if lines 2 and 5 are swapped? Case 2: What will be the output if line 3 is removed?

3. Given two non-empty doubly linked lists each representing a non-negative large integer such that each node represents one digit of the integer, write an algorithm to add these two integers. The head node represents the most significant digit of the integer. The result should be stored in a third linked list. Derive the time complexity of the proposed solution.

For the resulting list, the i-th list node (i≥1) is to be swapped with the (n-i+1)-th list node where n is the number of nodes in the list. Write an algorithm to achieve this functionality. Also, derive the time complexity.

4. Convert the infix expression (A + B)*(C $ (D - E) + F) - G to postfix using a stack. Draw a table to show the contents of the stack at every character read (Here $ is the exponentiation symbol). Also, evaluate the generated postfix expression for A = 2, B = 3, C = 4, D = 5, E = 1, F = 2, and G=6. Which data structure will you use for evaluating the postfix expression? For the given infix expression, create an expression tree. Use the expression tree to find the prefix notation. Further, reverse the given string (ignoring the parenthesis) using a stack and a queue without using any extra variable.

5. Consider the following recursive function:

```
Mystery(Integer) {
        if (Integer == 0)
                return 0
        if (Integer == 1)
                return  1
        else {
                x = Integer % 2 + 10 * (Mystery(Integer / 2))
                return x
        }
}
```

What function is **Mystery()** performing? What is the value of **Mystery(15)**? How many times will **Mystery()** be called to compute **Mystery(15)**? Also, draw a tree to show all generated calls. Show the activation record at every recursive call.
Write an iterative version of the function **Mystery** with the same functionality. Also, derive its computational complexity.

6. A 2-dimensional array **tray[m][n]** is used to represent an egg tray in which eggs are stored in m rows and n columns. The variants of egg trays available in the market have rows ranging from 1 to 6 and columns ranging from 1 to 9. Consider a scenario where a person buys 25 eggs. Identify dimensions of the two-dimensional array required to store the eggs. Also, develop a row-major and column-major mapping function that maps the resulting two-dimensional array into one-dimension.

Further, suppose 20 out of the 25 eggs break due to mishandling. Will you still prefer to use the tray of dimensions identified above? If no, determine the new dimensions required and justify with reasons.

Consider the arrangement of eggs in the **tray** to store 5 eggs at positions - tray[0][0], tray[1][1], tray[2][2], tray[3][3], and tray[4][4]. Modify the mappings developed above to represent it in one-dimension.