Data Structures (IT205)
First Midsemester-semester Exam

Time: 2 hours **marks: 60**

1. Binary Search works on a sorted set (sequence) of numbers from a totally ordered set (typically stored in an array). You may assume that all the key values are distinct.

   Thus, the fact that the list is in (increasing) sorted order is assumed in the code for implementing binary search. Thus, if the set of elements assumed to be sorted is indeed not in sorted error, the binary search algorithm may accidentally report an element missing. Note, however, that the binary search algorithm working on a set assumed to be sorted, will never report the presence of a non-existent number on the list. Thus, it potentially makes an error only in one direction.

   (a) List the set of **ALL** elements which binary search will fail to detect (despite their presence) in the following ordered sequence.
   $\{19, 25, 11, 3, 100, 6, 44\}$

   [5 marks]

   [Binary search compares the key value being searched for, with the element present in the middle of the subarray being searched. The mid-position is given by the expression $\lfloor \frac{L+U}{2} \rfloor$. Here, L and U are the left and right extreme positions of the subarray in which the search is being carried out]

   (b) Given an arbitrary set of n distinct elements in some order, we know that binary search will make no mistake if it is sorted. Assuming that the list may not necessarily be sorted, what is the largest possible number of elements (present in the n element list) which will fail to be detected by binary search. Is the sequence achieving this maximum error unique?

   [5 marks]

   (c) If not, then derive as good a lower bound as you can on the number of such sequences. (This means you guarantee that at least this many sequences produce maximum errors.)

   [5 marks]

2. You are provided with three stacks A, B and C. They are implemented in the standard way using an array and all of them have identical capacity of N. Initially Stack A has $n(< N)$ elements and the other two stacks are empty. You need to design a procedure to transfer the contents from Stack A to stack B using the basic stack operations of PUSH and POP. You may use the Stack C for temporary storage if required. No other memory is allowed. In addition, at no intermediate stage can two elements be present on any of the three stacks in reverse order from their original positions in the Stack A. That is, if element e_1 was higher up the stack then element e_2, initially, in Stack A, then at anytime if e_1 and e_2 are on the same stack, e_1 must be higher up in that stack than e_2.

   [**Hint**: Think of a recursive algorithm]

   [15 marks]

3. A binary search tree, in addition to having an underlying rooted binary tree structure, has key values which satisfy the **key value property** which states that the key value at a node is greater than the key value at any node in its left subtree and less than the key value at any node in its right subtree.

   Consider the following **modified** *key value property*. The key value at any node is greater than the key value of its left child (notice there is no requirement on the relative key values of the node with the entire left subtree, but only the left child). Analogously the key value at a node is less than the key value at its right child (and no condition on relative values with other nodes in the right subtree).

(a) Draw a binary tree with key values at the nodes, such that the tree satisfies the **modified** definition, but **does not** conform to the more conventional definition, presented first above.

[5 marks]

(b) Given a *modified binary search tree* as defined above, which node could be the maximum element in the whole tree? Give a precise description of the node.

[5 marks]

(c) Describe an algorithm for finding the maximum element in such a tree and analyse its running time.

[5 marks]

4. Compare the two functions $f_1(n) = n \log n$ and $f_2(n) = \log(n!2^n)$ in terms of their relative asymptotic growth rates. Express your answers in terms of the approriate operator among $o, O, \theta, \Omega, \omega$, and justify your answer. Here the notation $n!$ stands for the factorial of n.

[15 marks]