Data Structures (IT205)
Second Midsemester Exam
23^{rd} October, 2012

Time: 2 hours **marks: 80**

This question paper consists of 4 questions printed on 2 pages back-to-back. Please ensure your question paper is complete. Attempt ALL questions.

1. Suppose T is a Binary Search Tree with 30 nodes storing 30 **distinct** keys. Suppose the left-child of the root has the key value of rank 10 (the tenth smallest element) and the right-child of the root has the key value of rank 20 (the twentieth smallest element).
   (a) What are the possible values of the rank of the root.
   (b) What are the possible values for the number of nodes in the left subtree?
   (c) What is the maximum and minimum possible height of this binary search tree?
   (d) If a left-rotate is performed at the root then what are the possible values for the above three questions?

   [5+5+5+5 marks]

2. (a) For what values of $n \geq 1$ can you have red-black trees with exactly n key values, such that the tree has **zero** red nodes?
   (b) For a red-black tree with exactly n keys, what are the possible values for the number of red nodes?

   [10+10 marks]

3. (a) Draw 2-3-4 trees with 3 levels (height=2) with the largest and smallest possible number of keys.
   (b) If the block size of secondary storage is 512 bytes, and each record of elements stored in a B-Tree is of size 26 bytes. The common memory required to store properties of the whole node like the bolean variable indicating whether it is a leaf or not, the number of key values, parent etc require 18 bytes. The children pointer space is accounted for within the individual data records. What value of t will you choose to implement the B-Tree? A single node should not exceed the size of one block, and each block should retrieve as many records as possible, so that calls to load from secondary storage are minimised.

   [10+10 marks]

4. Suppose you have two arrays $A[1..n]$ and $B[1..n]$ such that each of them is a maximum binary heap. You also have an array $C[1...2n]$ available to you. Consider the following procedure which attempts to build a single maximum heap of the $2n$ elements of A and B and stores it in C.

```
1.if (A[1] > B[1])
2.   then C[1] ← A[1]
3.        LeftSubtreeArray ← B
4.        RightSubtreeArray ← A
5.        IndexLeft ← 2
6.        IndexRight ← 1
7.   else C[1] ← B[1]
8.        LeftSubtreeArray ← A
9.        RightSubtreeArray ← B
10.        IndexLeft ← 2
11.        IndexRightB ← 1
12.For i ← 2 to 2n
13.    level ← ⌊log2 i⌋
14.    if (i − 2^level + 1 ≤ 2^(level−1))
15.       then C[i] ← LeftSubtreeArray[IndexLeft]
16.            IndexLeft ← IndexLeft + 1
17.       else C[i] ← RightSubtreeArray[IndexRight]
18.            IndexRight ← IndexRight + 1
```

(a) Explain briefly what this code is doing.

(b) This code is incorrect, because it works only for some values of n. For each $n \leq 10$, state whether this code works correctly for that value of n or not.

(c) Give a general expression for those values of n for which this code works correctly

[6+6+8 marks]