ESO207A: Data Structures and Algorithms	*July-Nov 2017*
Mid-Semester Exam	*20th September, 2017*

Instructions. 1. For questions in which algorithms are asked for, your answer should provide, (a) a clear description of the algorithm in English and/or pseudo-code, (b) a correctness proof, and (c) an analysis of the running time of the algorithm. Marks will be deducted for imprecise and unclear descriptions or arguments.
2. You may reference and use algorithms and their properties as described in the class or homework solutions.

Problem 1. For problems requiring algorithms, give clear pseudo-code and explain the correctness and complexity arguments briefly but clearly. (6+6+6+7 = 25)

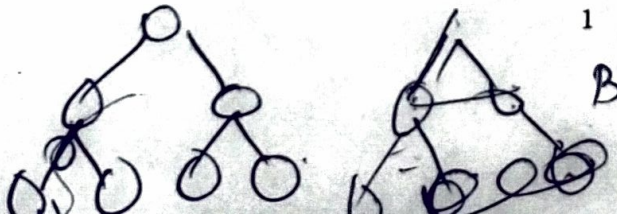
1 (a) Give reasons whether any array of 7 numbers can in general be sorted by a *comparison-sort* algorithm using 12 comparisons. ($2^8 = 256, 2^9 = 512, 2^{10} = 1024, 2^{11} = 2048, 2^{12} = 4096, 2^{13} = 8192, 2^{14} = 16384$).

1 (b) You are given a segment of a red-black tree shown in Figure 1 that has resulted from the deletion of some node and is in violation of red-black color properties. Show how to fix this violation using at most one rotation and by possibly changing colors of some nodes. Darkly shaded nodes are black. Node A is doubly black and node D is colored red. The node B has color c and node E has color c', where, $c, c' \in \{\text{RED}, \text{BLACK}\}$.

Figure 1: An RB-tree fragment in violation of RB-tree coloring properties.

1 (c) Suppose you are given a randomized procedure PARTITION(A, p, r) that takes an array segment $A[p, \ldots, r]$ of numbers and returns two indices q and s such that $p \le q \le s \le r$ such that $A[p, \ldots, q-1]$ are all strictly less than $A[q]$; $A[q], \ldots, A[s]$ are all equal; and $A[s+1, \ldots, r]$ are all strictly greater than $A[q]$. Write a recursive definition of the function SELECT(A, p, r, i) that returns the ith smallest element in $A[p, \ldots, r]$, for any given i, where, $1 \le i \le r - p + 1$.

1 (d) Given a binary tree on n nodes, outline an $O(n)$ time procedure that prints a leaf node that is furthest away from the root node (in terms of number of number of edges). Briefly argue correctness of the algorithm.



Problem 2. Non-dominated points (25)

A two-dimensional point (x, y) is said to dominate another two-dimensional point (u, v) if $x \geq u$ and $y \geq v$. Given a set P of points, a point $p = (x, y)$ is said to be a non-dominated point of P (also called a maximal point) if no other point q in P dominates p. See Figure 2 for examples. Given a set of n points P placed in arbitrary order in an array, give a time efficient algorithm FIND_NON_DOM(P, n) to find the set of all non-dominated points in P. *Notes*:

1. For simplicity, assume that no two points have the same x-coordinate or the same y-coordinate.
2. A point p is represented as a structure with two attributes $p.x$ and $p.y$. The set of points P, is represented as an array $P[1, \ldots, n]$ of points in arbitrary order.
3. You can find the index of the median of the points in $P[k, \ldots, l]$ by the x coordinate by using an informal statement like "let i = MEDIAN(P, k, l) by x-coordinate" Similarly, a statement like "let i =MEDIAN(P, k, l) by y coordinate" can be used to find the index of the median of the points in $P[k, \ldots, l]$ by the y coordinates. These functions run in time $O(k - l + 1)$. The median of n points is returned as the $\lfloor (n+1)/2 \rfloor$ th ranked item.
4. Full marks will be provided for a correct solution that takes $O(n \log n)$ time.
5. A correct solution of $O(n^2)$ time will earn only 10 points in total.

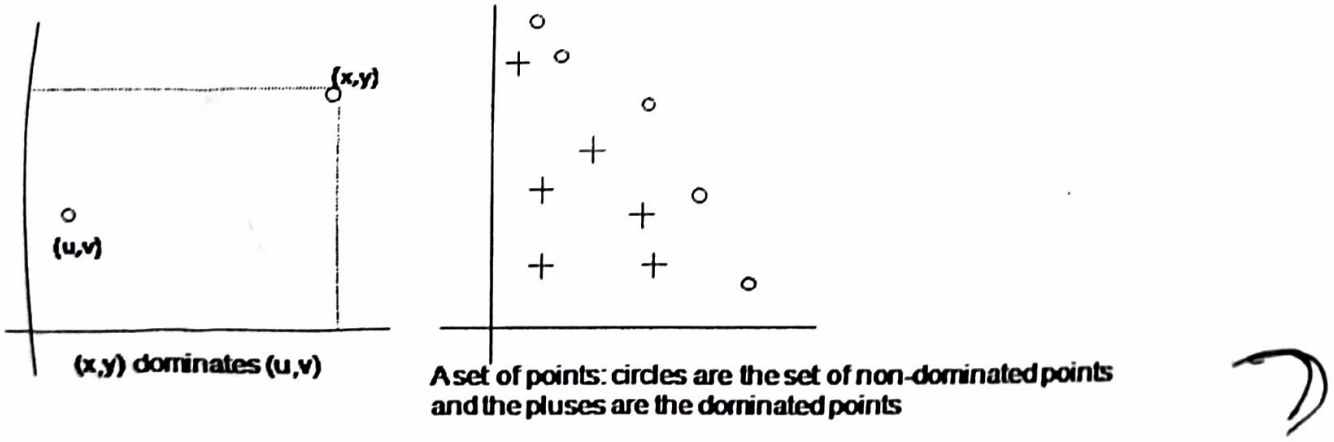


Figure 2: Dominated and non-dominated points in a point-set

Problem 3. Stack *Algorithm and Correctness: 20 Complexity 5*

You are given a sequence S consisting of the characters left parenthesis '(' and right parenthesis ')' only and of ***even*** length. Find the smallest number of parenthesis reversals that are needed to make the sequence a balanced parenthesis expression. Examples

$S_1 =)($ Output = 2
$S_2 =))((($ Output = 2
$S_3 =))(($ Output = 3

Explanation:

1. Write $S_1 =)_1(_2$ where the subscripts are used to label the parenthesis characters. No balanced parenthesis can start with) so this must be reversed to $(_1$. This gives $(_1(_2$ which requires a reversal of $(_2$ to $)_2$ to obtain $(_1)_2$ which is balanced.

2. Let $S_2 =)_1(_2)_3(_4$. The sub-expression $(_2)_3$ is balanced and can be removed giving $)_1(_4$ which is the same as S_1 and requires two reversals.

3. Write $S_3 =)_1(_2(_3(_4$. First reverse $)_1$ to $(_1$ as above. This gives the sequence $(_1(_2(_3(_4$. Then, $(_3$ and $(_4$ each can be reversed to give the sequence $(_1(_2)_3)_4$ which is balanced. (Also $(_2$ and $(_4$ can be reversed to give the sequence $(_1)_2(_3)_4$. Either way requires 3 reversals).

(*Notes*: The subscript numbering is shown only for convenience.)

Problem 4. Range queries over Red-Black Trees. A *range-query* RANGE_QUERY(T, a, b) takes a red-black tree T (with root node T) and keys a and b such that $a \leq b$, and returns *all* the nodes x of T such that $a \leq x.key \leq b$. The output order of the nodes is according to ordering in the inorder traversal of T. Assume T has n nodes in total.

1. Write an algorithm for CLOSEST_GE(T, a) that runs in time $O(\log n)$ and returns the leftmost node x in the inorder traversal of T such that $a \leq x.key$.
(*Algorithm + Correctness*: 12, *Complexity*: 3)

2. Consider the following algorithm for RANGE_QUERY(T, a, b) that uses CLOSEST_GE(T, a) as a sub-routine and the SUCCESSOR(T, x) function on binary search trees. Show that this simple algorithm runs in time $O(m + \log n)$ for an n-node red-black tree, where, m is the number of nodes output. (10)

```
RANGE_QUERY(T, a, b)
1.  x = CLOSEST_GE(T, a)
2.  while x ≠ NIL and x.key ≤ b
3.      print x
4.      x = SUCCESSOR(x)
```

For reference, the SUCCESSOR function is given below.

```
SUCCESSOR(x)
1.  if x.right ≠ NIL
2.      return MINIMUM(x.right)
3.  else
4.      while x.p ≠ NIL and x == x.p.right
5.          x = x.p
6.      return x.p
```