Assignment 3

Answer all questions – maximum 100 marks. You must score at least 50 to pass the assignment.

1. (5 marks) Illustrate that the nodes of any AVL tree T can be colored “red” and “black” so that T becomes a red-black tree.

2. (5 + 10 = 15 marks) Illustrate that via AVL single rotation, any binary search tree T1 can be transformed into another search tree T2 (with the same items) (5 marks).
 Give an algorithm to perform this transformation using *O*(N log N) rotation on average (10 marks).

3. (10 + 2 = 12 marks) Suppose you are given two sequences S1 and S2 of *n* elements, possibly containing duplicates, on which a total order relation is defined. Describe an efficient algorithm for determining if S1 and S2 contain the same set of elements (10 marks).
 Analyze the running time of this method (2 marks).

4. (5 + 8 = 13 marks) Given sequence 3, 1, 4, 1, 5, 9, 2, 6, 5, 3, 5, sort the sequence using the following algorithms, and illustrate the details of the execution of the algorithms:

   a. (5 marks) merge-sort algorithm.

   b. (8 marks) quick-sort algorithm. Choose a partitioning strategy you like to pick a *pivot* element from the sequence. Analyze how different portioning strategies may impact on the performance of the sorting algorithm.

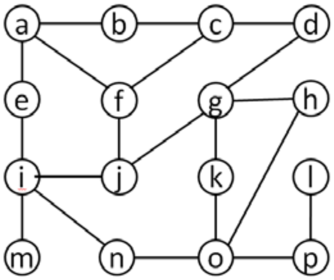
5. (4 + 4 + 7 + 10 = 25 marks) Given the graph shown below, answer the following questions:

   a. (4 marks) Illustrate the sequence of vertices of this graph visited using depth-first search traversal starting at vertex **g.**

   b. (4 marks) Illustrate the sequence of vertices of this graph visited using breadth-first search traversal starting at vertex **b.**

   c. (7 marks) Illustrate adjacency list representation and adjacency matrix representation, respectively, for this graph. What are the advantages and disadvantages of those two representations?

   d. (10 marks) Describe an algorithm to find in the graph a path illustrated below that goes through every edge exactly once in each direction.



6. (10 marks) Exercise 9.7. Why does the method remove(x) in the `RedBlackTree` implementation perform the assignment `u:parent = w:parent`? Shouldn't this already be done by the call to `splice(w)`?

7. (10 marks) Exercise 10.8. Implement the `remove(u)` method, that removes the node `u` from a `MeldableHeap`. This method should run in *O*(log n) expected time.

8. (10 marks) Exercise 11.12. Prove that a binary tree with *k* leaves has height at least log *k*.