CMPT 280: Intermediate Data Structures and Algorithms

Practice Final Examination

Dislcaimer: These are some samples of questions that have appeared on past final exams. It is not an exhaustive sampling of questions you might encounter on this year's exam. Other types of questions might occur. Nor is it necessarily representative of how many questions will be on the final exam or of the coverage of topics. Completing the practice final is not a substitute for studying though it may help inform the things you are most in need of studying.

1. (1 point) Exactly how many statements are executed by the following code? Assume that the function `allons_y()` is $O(1)$ and therefore counts as one statement.

```
for(int j=0; j < n; j++) {
   for(int k=0; k < j; k++) {
      allons_y(j,k);
   }
}
```

   A. $n^2 + 2n$ B. $n^2 + 1$ **C.** $n^2 + n + 1$ D. $2n + 2$

2. (1 point) Does there exist a function $f(n)$ for which the code from the previous question is $\Theta(f(n))$?

   A. Yes. B. No.

3. (1 point) The function $t(n) = 3n \log 5n + 8\sqrt{n} + 12n^{5/2} + 100$ is:

   A. $O(n \log n)$

   B. $O(\sqrt{n})$

   C. $O(n^{5/2})$

   D. All of a), b), and c) are true.

   E. None of a), b), and c) are true.

   (If you think exactly one of A, B, or C is true, choose A, B, or C. Otherwise, choose one of D or E.)

4. (1 point) The function $t(n) = 12n + 42 \log^2 n + 19n \log n + n^3 + 87$ is:

   A. $O(n^3)$

   B. $O(n^4)$

   C. $\Theta(n^3)$

   D. $O(2^n)$

   E. All of a), b), c), and d) are true.

   F. None of a), b), c), and d) are true.

   (If you think exactly one of A, B, or C is true, choose A, B, or C. Otherwise, choose one of D or E.)

5. (1 point) True or false? A linked list of n integers occupies more memory than an array of n integers.

   A. True. B. False.

For questions 6 to 9 suppose you have been given the task of designing and implementing a B-tree ADT. Each question will ask you whether a particular decision should be made at the time of writing the specification for the B-Tree (specification time) or delayed until the time when the B-Tree is being implemented (implementation time).

6. (1 point) The decision of what order(s) of B-tree will be supported should be made at:

   A. Specification time. B. Implementation time.

7. (1 point) The decision of whether the B-Tree ADT should extend another ADT should be made at:

   A. Specification time. **B. Implementation time.**

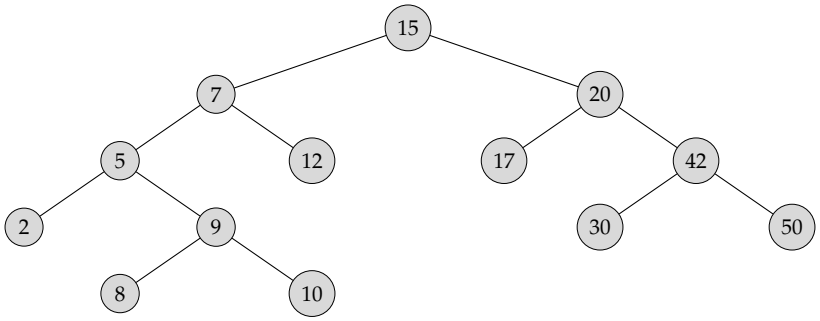
8. (1 point) The decision of how to store the children of a given node in the B-tree should be made at:

   A. Specification time. **B. Implementation time.**

9. (1 point) Selection of the data types of the parameters to the operation of looking up an element in the B-tree should be made at:

   A. Specification time. **B. Implementation time.**

For questions 10 through 13, please refer to the following tree:



10. (1 point) For a **breadth-first** traversal of the above tree, starting from the root, which of the following statements is true?

    A. Nodes 7, 17, and 42 may be visited in any order.

    B. Node 7 must be visited before node 17 and node 17 must be visited before node 42.

    C. Node 7 must be visited after both nodes 17 and 42, but nodes 17 and 42 may be visited in either order.

    D. Node 7 must be visited before both nodes 17 and 42, but nodes 17 and 42 may be visited in either order.

11. (1 point) For a **depth-first** traversal of the above tree, starting from the root, if the first two nodes visited are 15 and then 20, which of the following statements is true?

    A. Nodes 5, 42, and 30 may be visited in any order.

    B. Node 42 must be visited before node 30, and node 5 must be visited before both nodes 42 and 30.

    C. Node 42 must be visited before node 30, and node 5 must be visited after both nodes 42 and 30.

    D. Node 30 must be visited before node 42, and node 5 must be visited after both nodes 42 and 30.

12. (1 point) Which one of the following statements about the above tree is true?

A. It is a binary tree.

B. It is both a binary tree and an ordered binary tree (binary search tree).

C. It is a binary tree, an ordered binary tree, and an AVL tree.

D. It is a binary tree, and an AVL tree, but not an ordered binary tree.

13. (1 point) Which of the following arrays is a correct arrayed representation of the above tree?

A.

-	15	7	5	2	9	8	10	12	20	17	42	30	50

B.

-	15	7	20	5	12	17	42	2	9	30	50	8	10

C.

-	2	5	7	8	9	10	12	15	17	20	30	42	50

D. None of the above.

14. (1 point) Which of the following arrayed representations of trees describes a tree that is also a heap?

A.

-	42	39	40	27	19	11	16	8	20	21

B.

-	42	39	40	27	19	11	16	22	20	17

C.

-	42	39	40	19	27	11	16	8	20	21

D. None of the above are valid arrayed heaps.

15. (1 point) The time complexity of the insertion and deletion algorithms for ordered binary trees is:

A. $\Theta(n)$ B. $\Theta(n \log n)$ C. $\Theta(\log n)$ D. $\Theta(n^2)$ **E. None of the above.**

16. (1 point) Which of the following is **not** a property of a 2-3 tree as we defined them in class?

A. Every internal node has exactly two or three children.

B. Every internal node contains a key-element pair.

C. Every leaf node contains a key-element pair.

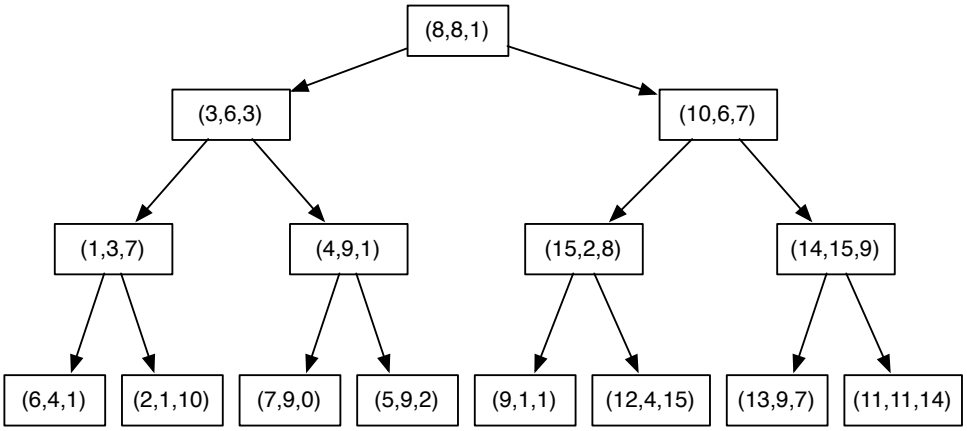
D. All of the leaf nodes are at the same level.

E. Keys-element pairs with keys that are smaller than the key of an internal 2-node (a node with two children) are found in the left-most subtree.

17. (1 point) In the worst case, how many nodes will be examined when searching a 2-3 tree for a particular key-element pair if the tree contains n key-item pairs?

A. $O(n^2)$ B. $O(n \log n)$ **C.** $O(\log n)$ D. $O(n)$

18. (1 point) Given the following 3-D tree, which set of nodes are recursively **visited** by the range search algorithm if the query range has corners (1, 3, 6) and (3, 5, 9)?



A. **(8,8,1), (3,6,3), (1,3,7), (6,4,1), (2,1,10)**

B. (8,8,1), (3,6,3), (1,3,7), (6,4,1)

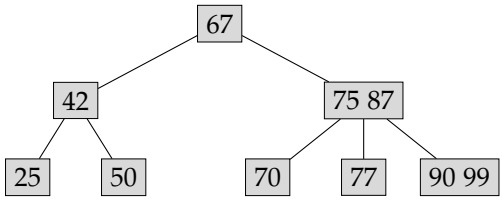
C. (8,8,1), (3,6,3), (1,3,7), (6,4,1), (2,1,10), (10,6,7), (15, 2, 8), (9, 1, 1)

D. Every node in the tree is visited.

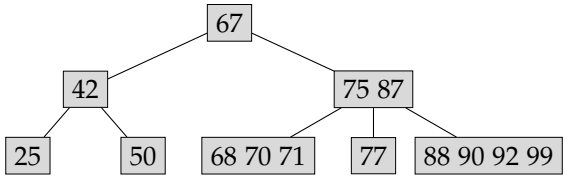
19. (1 point) For the *k*-D tree and range search described in question 18, how many points are actually **returned** by the range search?

A. 0 **B.** **1** C. 4 D. 8 E. All of them.

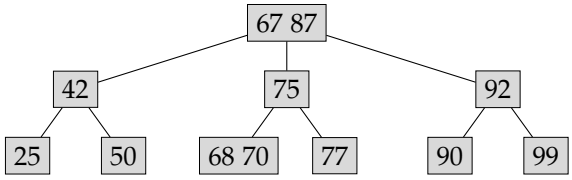
20. (1 point) Consider the following B-tree of **order 2** (nodes in a B-tree of order b can have between b and $2b$ children except for the root which can have between 2 and $2b$ children):



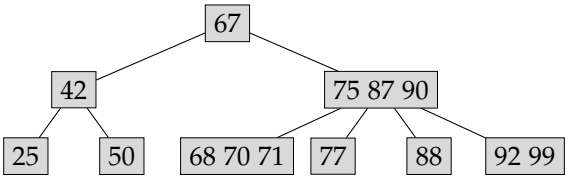
Which of the following trees results form inserting the keys 68, 92, 88, 71 in that order?



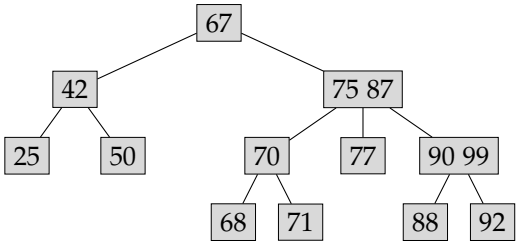
A



B



C



D

Solution: The correct answer is C.

21. (1 point) Which one of the following statements is **not** true?

A. B-trees contain key-element pairs in their internal nodes.

B. The leaf nodes of both B-trees and B+ trees may contain more than one key-element pair.

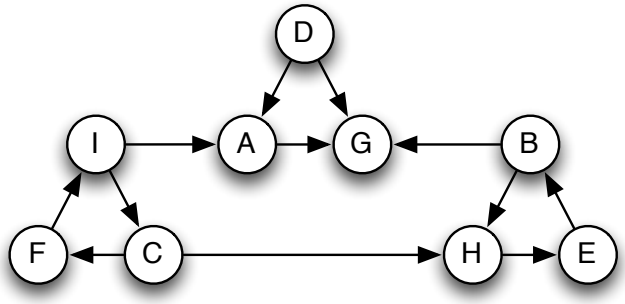
C. 2-3 trees are special cases of B+ trees.

D. The leaf nodes of both B-trees and B+ trees can be linearly linked (like a linked list) from left to right to permit fast, sequential, key-order access to elements.

22. (1 point) Recall that the *load factor* of a hash table is equal to the number of items in the hash table divided by the number of hash table entries (buckets). In other words, the number of load factor is the average number of items in each hash table entry. In a chained hash table where the buckets are implemented as linked lists, if the load factor of x, what will be the average number of **items** examined while searching for and retrieving a particular item?

A. $\log_2 x$ **B.** $x/2$ C. x D. $2x$ E. $10x$ F. independent of x

For questions 23 to 30, please refer to this graph:



23. (1 point) What is the indegree of node G?

    A. 1 B. 2 **C. 3** D. 4

24. (1 point) What is the outdegree of node D?

    A. 1 **B. 2** C. 3 D. 4

25. (1 point) The sequence of nodes C, F, I, C, H, E, B, G is a:

    A. A walk, but not a trail.
    B. A trail, but not a path.
    C. A path.
    D. A circuit, but not a cycle.
    E. A cycle.

26. (1 point) The sequence of nodes C, F, I, A, G is a:

    A. A walk, but not a trail.
    B. A trail, but not a path.
    C. A path.
    D. A circuit, but not a cycle.
    E. A cycle.

27. (1 point) The sequence of nodes E, B, H, E is a:

    A. A walk, but not a trail.
    B. A trail, but not a path.
    C. A path.
    D. A circuit, but not a cycle.
    E. A cycle.

28. (1 point) How many strongly connected components consisting of more than one node are in the graph?

    A. 1 **B. 2** C. 3 D. 4 E. 9

29. (1 point) Assume that the graph is represented as an adjacency list where the nodes in each adjacency list are in alphabetical order. Under this assumption, in which of the following orders would the nodes be visited during a **depth-first traversal** starting at vertex I?

A. I, A, G, C, F, H, E, B

B. I, F, I, A, G

C. I, A, F, G, C, H, E, B

D. I, C, F, H, E, B, A, G

30. (1 point) Assume that the graph is represented as an adjacency list where the nodes in each adjacency list are in alphabetical order. Under this assumption, in which of the following orders would the nodes be visited during a **breadth-first traversal** starting at vertex I?

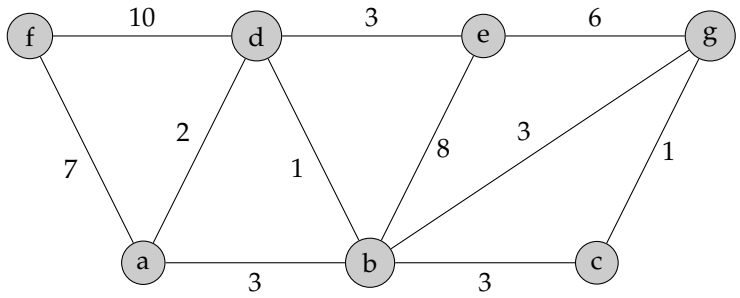
A. I, A, C, F, G, H, E, B

B. I, F, A, C, H, G, B, E

C. I, C, A, F, G, H, B, E

D. I, A, C, G, F, H, E, B

31. (1 point) Consider the following graph:



Which of the following are possible orders in which edges could be added to the minimum spanning tree of the above graph by Kruskal's algorithm?

A. [B,D], [G,C], [A,D], [D,E], [G,B], [A, F]

B. [B,D], [G,C], [A,D], [B,G], [D,E], [A, F]

C. [B,D], [G,C], [A,D], [B,C], [D,E], [A, F]

D. All of a), b), and c) are possible orders.

E. None of a), b), and c) are possible orders.

32. (1 point) What is the worst-case time complexity of Warshall's algorithm?

A. $O(n)$

B. $O(n^2)$

C. $O(n^3)$

D. $O(\log n)$

E. None of the above.

33. (1 point) Which of the following are valid inputs to Dijkstra's algorithm?

    A. Any unweighted directed graph.

    B. Any weighted graph, directed or undirected, with arbitrary weights.

    C. Any weighted graph, directed or undirected, with non-negative weights.

    D. All of the above.

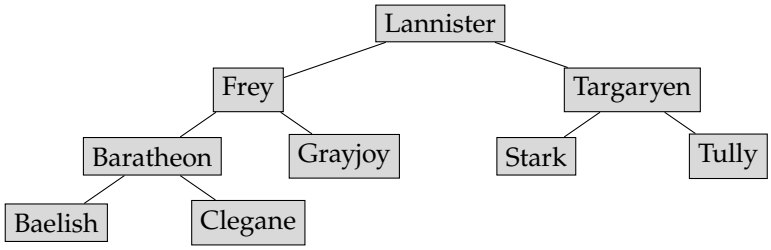
    E. None of the above.

34. (1 point) True or false? The operation of finding all edges incident on a node v in a graph that is implemented using an adjacency matrix is $O(degree(v))$.

    A. True **B. False**

The remaining questions are written, short-answer style questions. Answer these questions directly on this question paper in the spaces provided.

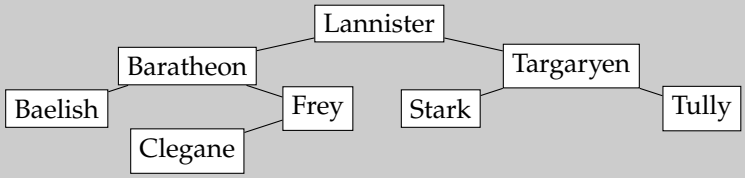
35. Consider the following AVL tree which is keyed on strings in dictionary (lexicographic) order.



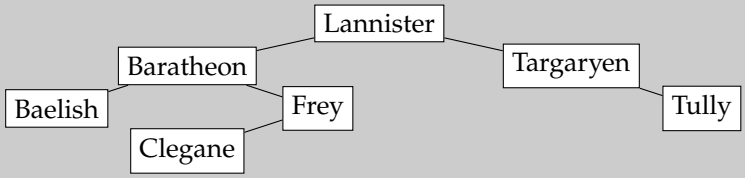
George R. R. Martin is managing his database of characters and has killed off so many characters in his books that he needs to remove entire Houses from his database. Delete the names Grayjoy, Stark and Baelish from the above AVL tree in that order. **Draw the resulting tree** after each successive deletion and **write the names the rotations performed** in the course of each deletion (if any). (That is, delete Grayjoy, obtain the result, delete Stark from the result of the first deletion, and so on...your answer should consist of 3 AVL trees.)

Solution:

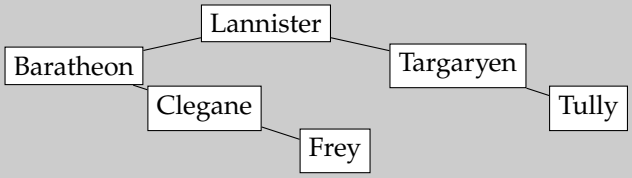
Delete Grayjoy, requires single right rotation (LL rotation) at Frey.



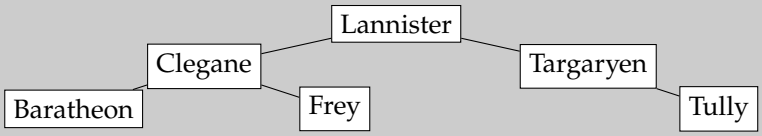
Delete Stark (no rotations required).



Delete Baelish, requires a double left rotation (RL rotation) at Baratheon. i.e. a right rotation at Frey...



followed by a left rotation at Baratheon:



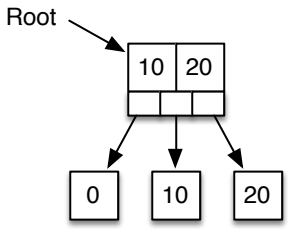
36. (8 points) Suppose the java class `ExpressionTree` implements an expression tree. Recall that an expression tree represents an arithmetic expression and stores arithmetic operations in its internal nodes, and operands (numbers) in its leaf nodes. **Write a recursive java method** for `ExpressionTree` which, given the root node of an expression tree as a parameter, evaluates the expression using a **post-order traversal** and returns the value of the expression stored in the expression tree. Assume that the tree supports only addition, subtraction, multiplication, and division, and that leaf nodes store floating-point values. Assume that tree nodes are objects of type `ENode`. Additionally assume that:

- `ENode` has an `isLeaf()` method that returns true if a node is a leaf.
- `ENode` has a `getValue()` method that return the floating-point number contained within if the node is a leaf node.
- `ENode` has a `getOperator()` method that return a `char` indicating which operator is stored within (`+`, `-`, `*`, `/`) if the node is an internal node.
- `ENode` has methods `leftChild()` and `rightChild()` that return references to the left and right child nodes, respectively.

Solution: Should look something like this. `float` in place of `double` is OK. Basically, a post-order traversal. There could include some checks for error cases, but those are not required.

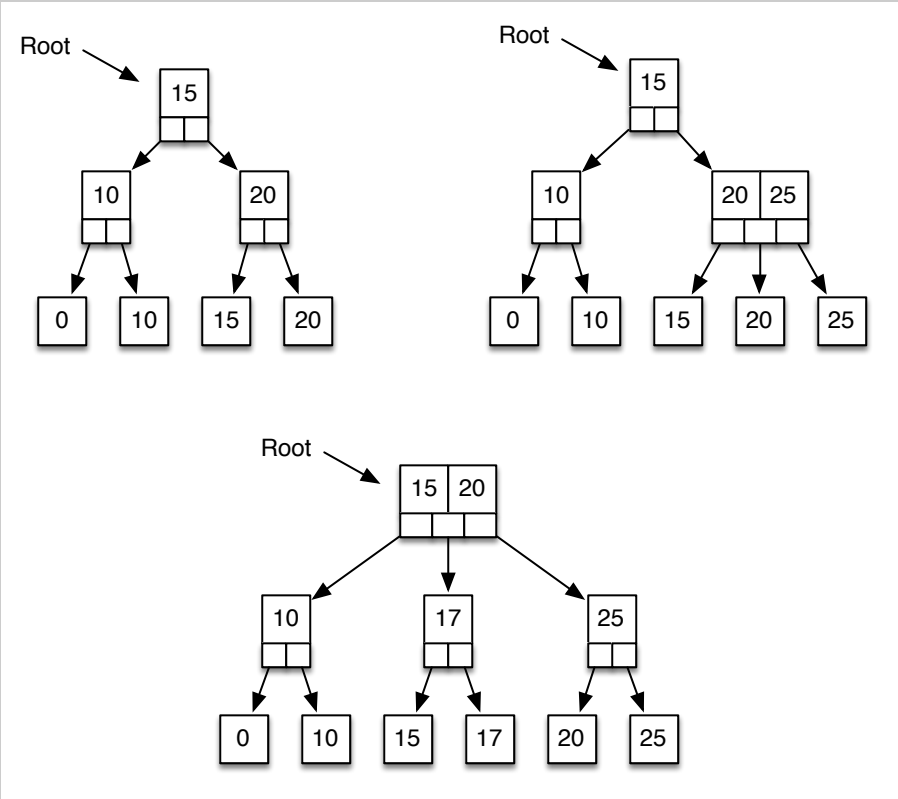
```
protected double evaluate(ENode root) {

      // If this is a leaf (an operand) return its value.
      if( root.isLeaf() )
         return root.getValue();

      // Otherwise, it must be an internal node.

      // Process left, process right...
      double leftOperand = evaluate(root.leftChild());
      double rightOperand = evaluate(root.rightChild());

      // Process root.
      switch(root.getOperator()) {
      case '+':
         return leftOperand + rightOperand;
      case '-':
         return leftOperand - rightOperand;
      case '/':
         return leftOperand / rightOperand;
      case '*':
         return leftOperand * rightOperand;
      }
   }
```

37. (6 points) Consider the following 2-3 tree:

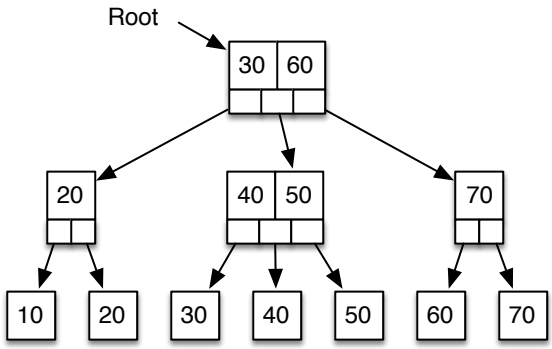


Beginning with the 2-3 tree above, draw the resulting 2-3 tree after insertion of **each** of the following values: 15, 25, and then 17. Insert the second number into the tree resulting from the first insertion, and so on; you should have three trees drawn for your answer.

Solution:

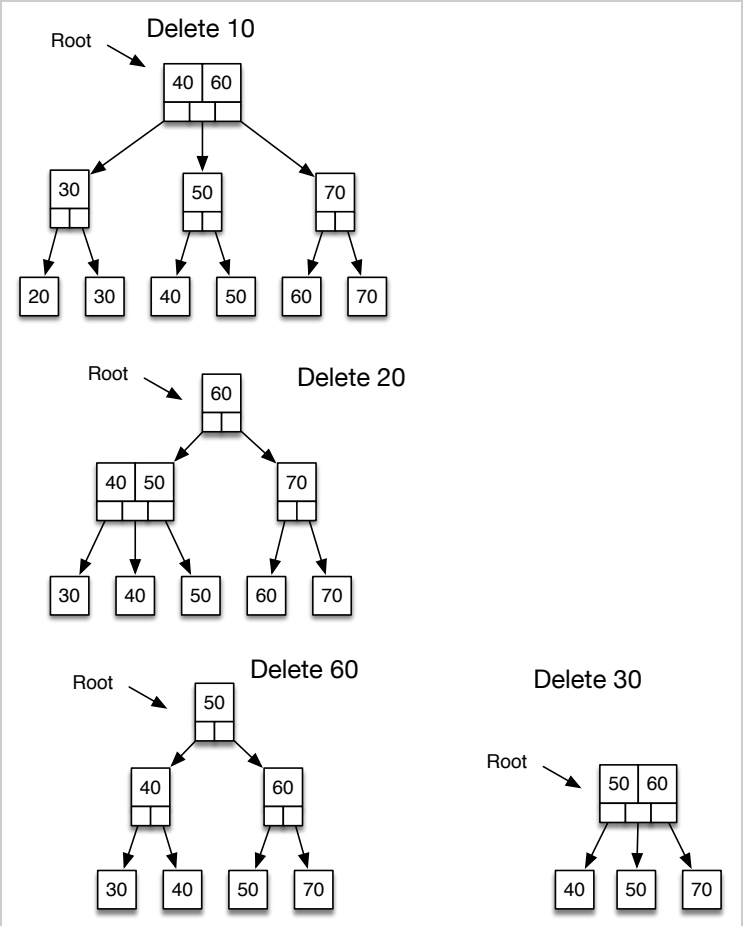


38. (8 points) Consider the following 2-3 tree:



Beginning with the 2-3 tree above, draw the resulting 2-3 tree after deletion of **each** of the following values: 10, 20, 60, 30. Delete the second number from the tree resulting from the first deletion, and so on; you should have four trees drawn for your answer.

Solution:

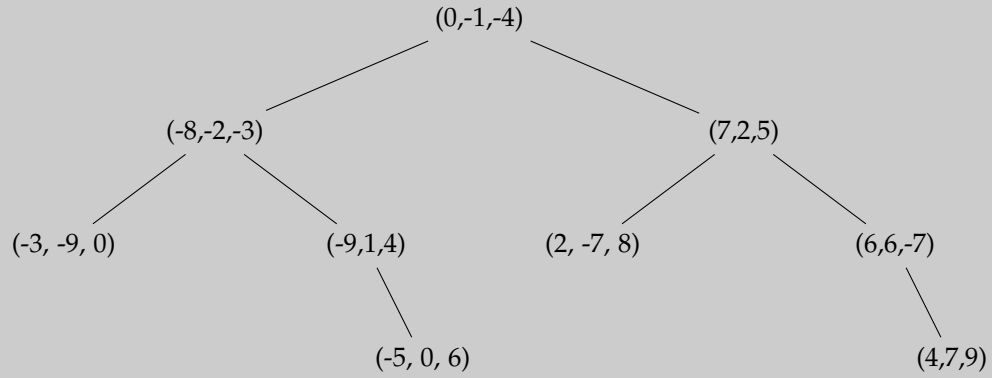


39. This set of questions concerns kd-trees.

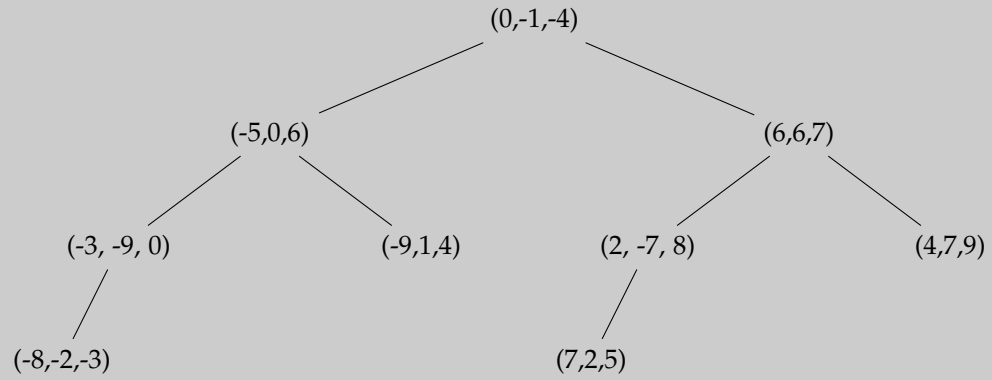
(a) (6 points) Construct a balanced kd-tree from the following set of points and draw the resulting kd-tree (only the final answer is required, but if you show some or all of your work, you will be more likely to get partial marks if you make a mistake).

(4, 7, 9)	(2, -7, 8)	(-9, 1, 4)
(7, 2, 5)	(-8, -2, -3)	(0, -1, -4)
(6, 6, -7)	(-3, -9, 0)	(-5, 0, 6)

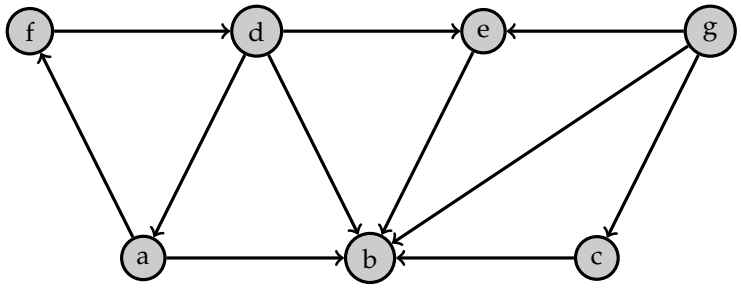
Solution: Using the convention of choosing the "lower middle" when there are an even number of elements to partition:



Using the convention of choosing the "upper middle" when there are an even number of elements to partition:



40. Consider the following graph:



(a) (3 points) Draw an adjacency matrix representation of this graph.

Solution:

0	1	0	0	0	1	0
0	0	0	0	0	0	0
0	1	0	0	0	0	0
1	1	0	0	1	0	0
0	1	0	0	0	0	0
0	0	0	1	0	0	0
0	1	1	0	1	0	0

(b) (3 points) Draw an adjacency list representation of this graph.

Solution:

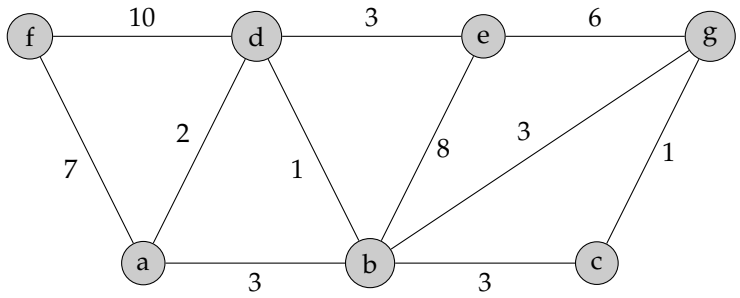
```
The lists for each node in order from a to g should be:

b, f
empty list
b
a, b, e
b
d
b, c, e

Nodes on each list can be in any order.
```

(c) (2 points) Give a Java **variable declaration** to define a data structure to store a graph adjacency list. You may assume `lib280` is available.

Solution: Should be an array of lists, along the lines of this, as long as it is an array of lists or list of lists of edges of some sort. It need not use lib280.

```
LinkedList280< Edge280<Vertex280> >[] adjList;
```

41. Recall that Dijkstra's algorithm used three arrays: `visited`, `tentativeDistance`, and `predecessor`. Consider the following graph:



Each row of the table below represents the three arrays after the processing of each node by Dijkstra's algorithm. Fill out the contents of each array showing their contents after each node has been processed in the order they would be processed by Dijkstra's algorithm **given that the start node is node g**. Also indicate in the left column which node was processed at that step. If there is ever more than one choice for the vertex to be processed next, choose the vertex closer to the beginning of the alphabet. The first row shows the initialization, before any nodes have been processed.

Node	`visited`							`tentativeDistance`							`predecessor`						
	a	b	c	d	e	f	g	a	b	c	d	e	f	g	a	b	c	d	e	f	g
-	F	F	F	F	F	F	F	∞	∞	∞	∞	∞	∞	0	-	-	-	-	-	-	-

42. Answer these questions about sorting. Provide justification for your answers.

    (a) In practice, which of merge sort or quick sort is faster for sorting random sequences of floating point numbers of arbitrary length?

    Solution: Since the sequences are random, the worst case for quick sort will be highly unlikely and most sorts will be close to the average case. Since quick sort has less overhead (same big-O time complexity, but smaller constant) than merge sort, quick sort will usually be faster.

    (b) From among the sorts we covered in class, what is the fastest sorting algorithm for strings?

    Solution: MSD radix sort. It's linear time and it sorts strings in the right order, unlike LSD radix srot.

    (c) What is the best-case time complexity of selection sort?

    Solution: $O(n^2)$ - you always have to do a linear search for the next-smallest element from among the elements remaining.

43. You can expect the last question on the final exam to be a fairly substantial programming question. You'll be given an ADT specification for an ADT that you haven't seen before and will need to implement the ADT in Java. You will be allowed to use the lib280 classes and built-in Java API classes, but it will be graded on the appropriateness of your choices to use or not use these resources, and, if you choose to use these resources, how you use them. You can expect the given specification to require the implementation of no more than five short-ish methods.