CS112 Data Structures - Midterm 2 - Fall 2022

Name: _____ NetID: _____

- WRITE your **Name** and **NetID** on EVERY page.
- DO NOT REMOVE the staple on your exam.
- WRITE NEATLY AND CLEARLY. If we cannot read your handwriting, you will not receive credit. Please plan your space usage. No additional paper will be given.
- This exam is worth 150 points.

Problem 1 – Binary Search Tree (BST) comprehension (28 points)

(a) **(5 points)** Build the BST whose keys are inserted in the following sequence.
35 20 25 50 17 73 45 19 37 3 18 48 36

The following questions refer to the BST on part (a).

(b) **(3 points)** What is the tree height of the BST (the root has height 0)?

(c) **(3 points)** List the keys that requires 4 compares (count compareTo) for a search hit.

Name: _____ NetID: _____

(d) **(5 points)** What could be the maximum height of a BST built with the same keys, but with different insertion sequences, and why?

(e) **(4 points)** For the cases that the BST with the maximum height, what could be the key in the root node?

(f) **(5 points)** What could be the minimum height of a BST built with the same keys, but with different insertion sequences, and why?

(g) (3 points) What is the average number of compares (count compareTo) for a search hit? Simply write the expression.

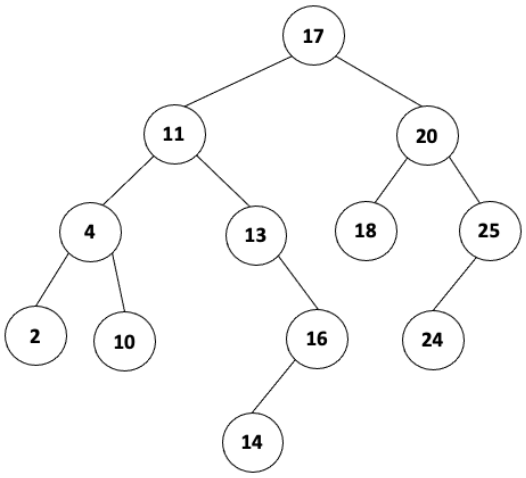
Name: _____ NetID: _____

Problem 2 – Binary Search Tree (BST) implementation (20 points)

(a) **(5 points)** Assume the print () operation below is provided in the BST implementation. Given the BST below, write the output when a client program calls the print() method. The root node is the node containing key 17.

```
public void print() {
    foo(root);
}

private void foo(Node x)
{
    if (x == null)
        return;
    StdOut.print(x.key);
    foo(x.left);
    foo(x.right);
}
```



(b) Assume the delete() method provided by the BST API is implemented based on the deletion discussed in class. Given the BST in (a), a client program calls the delete() method to remove the node with key **20**, and then calls the delete() method again to remove the node with key **13**.

a. **(10 points)** Draw the resulting tree after removing the 2 nodes.

b. **(5 points)** Write the output when the client program calls the print() method after removing the 2 nodes.

Name: _____ NetID: _____

Problem 3 – 2-3 Trees (30 points)

Construct a 2-3 tree whose keys are inserted in the following sequence.

5 25 10 4 30 2 9 3 28

(a) **(6 points)** Show the 2-3 tree after the insertions of 5 25 10 4 30

(b) **(6 points)** Show the 2-3 tree after the insertions of 5 25 10 4 30 2 9 3

(c) **(6 points)** Show the 2-3 tree after the insertion sequence completed.

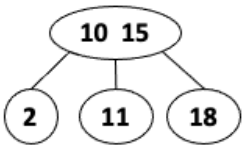
(d) **(6 points)** What is the number of links (perfect balance) of the tree from part (c)?

(e) **(6 points)** Briefly explain why we would use a 2-3 tree over a standard BST?

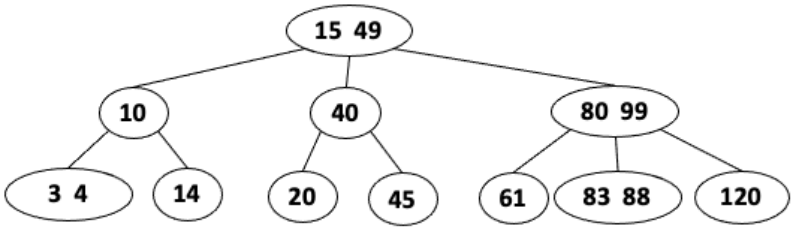
Name: _____ NetID: _____

Problem 4 – Left-Leaning Red-Black Tree (42 points)

a) **(5 points)** Draw the corresponding left-leaning red-black tree to the 2-3 tree below. Label red links R.



b) **(15 points)** Draw the corresponding left-leaning red-black tree to the 2-3 tree below. Label red links R.



Name: _____ NetID: _____

c) (5 points) What is the number of black links (perfect black balance) in the left-leaning red-black tree from (b) in any of the paths from the root to null links.

d) (7 points) What is the minimum height (tilde notation) of a LLRB tree in terms of *n*, where *n* is the number of items in the tree? Why?

e) (10 points) Assume the insert() method provided by the LLRB API is implemented based on the insertion discussed in class. Given the LLRB tree in (b), a client program calls the insert() method to insert the node with key **2**. Draw the resulting tree after the node is inserted.

Name: _____ NetID: _____

Problem 5 – Priority Queue (30 points)

a) **(5 points)** Is an array that is sorted in increasing order a MIN binary heap? Why?

b) Below is an array representing a valid binary heap for a MIN heap.

index	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
		2	3	9	7	5	12	20	7	11	20	14	18			

(i) **(12 points)** Show the array contents after 2 insertions: first insert key **6** and then insert key **1**.

index	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

(ii) **(13 points)** After the above insert operations, assume 2 (two) delMin() operations were performed, show the contents of the array.

index	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15