Midterm 02

COSC 211: Data Structures, Fall 2021

Instructions. This exam is open book and open note—you may freely use your notes, lecture notes, or textbook while working on it. You may *not* consult any living resources such as other students or web forums. The exam must be submitted by **5:00 PM on Thursday, November 11th, 2021**. If you do not attend class in person, you may email your scanned or typeset solution **in PDF format** to the professor using the subject line `[COSC 211] Midterm 02`.

In order to receive full credit, be sure to show your work for each problem (where applicable)—an answer without justification is not guaranteed to receive any credit.

Affirmation. I attest that that work presented here is mine and mine alone. I have not consulted any disallowed resources while taking this exam.

Name: ____

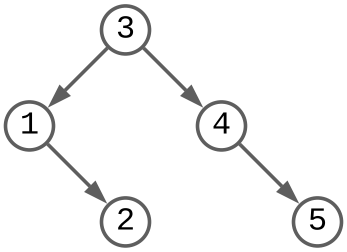
Signature: ____

Problem 1. Consider an `SSet` implementation using an *unbalanced* binary search tree (BST).

(a) Starting from an initially empty BST, in the space below draw the tree depicting the state of the BST after adding the elements `10, 15, 20, 5, 25, 3, 7, 12, 11, 16, 17` in that order (i.e., after calling `add(10)`, `add(15)`, `add(20)`, etc.).

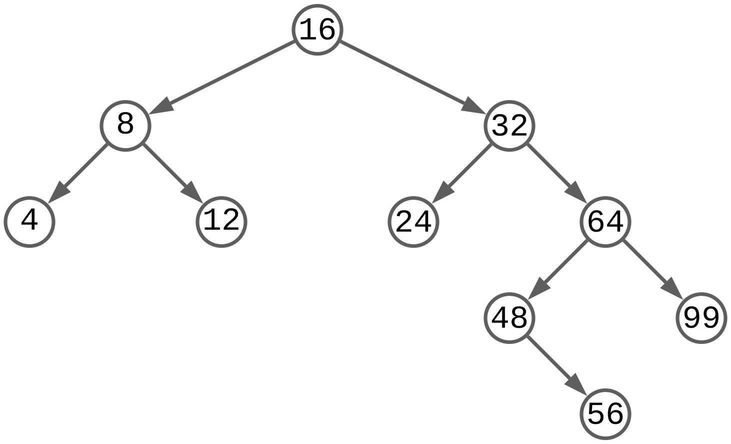
(b) In the space below, draw the resulting BST after performing the operation `remove(15)`.

(c) Starting from an empty BST, the following state is the result of adding the elements `3, 1, 4, 2, 5` in that order:



In the space below, list all orderings of the numbers `1` through `5` such that adding the elements in that order to an initially empty BST would result in the same state depicted above.

Problem 2. The following BST is *not* an AVL (height balanced) tree:



(a) On the tree above, indicate the height of each node, and find all nodes that are not height balanced.

(b) The AVL property of the tree above can be restored by performing a single restructure operation. In the space below, draw the (AVL) tree resulting from performing this restructure operation.

Problem 3. Consider the following binary heap, represented as an array:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	5	3	21	13	7	11	23	25	17	19	9	15		

(a) In the space below, express the heap above as a complete binary tree.

(b) What is the *array* resulting from the operation `add(2)`?

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

(c) *Starting from the original heap*, what is the array resulting from the `removeMin()` operation?

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

Problem 4. Consider the process of throwing two balls into 4 bins, labeled `1, 2, 3, 4`. That is, the two balls are thrown independently, and each ball is equally likely to land in each bin.
(a) Define a sample space representing the process above. What are the probabilities of the each outcome in the sample space?

(b) Suppose the first ball is thrown into bucket I and the second is thrown into bucket J. What is the probability that $J = I$? What is the probability that $J = I + 1$?

(c) Define the following game using the balls-into-bins random process: If both balls land in the same bin, you win \$2. If the two balls land in adjacent bins (i.e., $J = I + 1$ or $J = I - 1$), you in \$1. Otherwise, you lose \$1. What is the expected payout (winnings) from playing this game?