

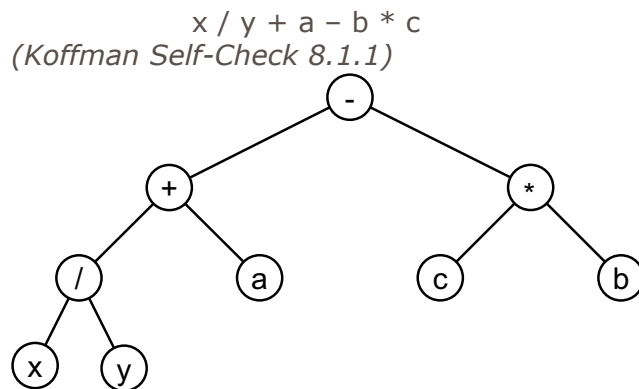
ITEC200 – WEEK08 CONCEPTUAL SOLUTIONS

This worksheet is comprised of some questions that students do as preliminary work and then some group-work activities that are completed in online classes. The first part of each online class is focused upon discussing the preliminary solutions that students have submitted. (Note that students must submit all preliminary work by 12pm the day before the online class by using the "Submit Preliminary Activities" link from the unit homepage.)

PRELIMINARY QUESTIONS

QUESTION 1

Draw a binary expression tree for the following infix expression. Your trees should enforce the Java rules for operator evaluation (higher-precedence operators before lower-precedence operators and left associativity).



QUESTION 2

Using the Huffman Tree in Figure 8.5 of the Huffman text (pg 402) decode the following binary string:

1100010001010001001011101100011111110001101010111101101001

(Koffman Self-Check 8.1.2)

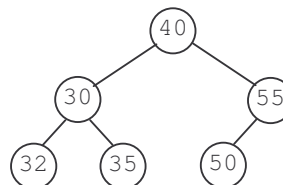
"where is waldo"

QUESTION 3

For the tree shown to the right:

- What is its height?
- Is it a complete tree?
- Is it a full tree?
- Is it a binary search tree?

(Koffman Self-Check 8.1.3)



- 3
- yes
- no
- no

QUESTION 4

If visiting a node in the tree from Question 3 displays the integer value stored, show the result of:

- a) an inorder traversal
- b) a preorder traversal
- c) a postorder traversal.

(Koffman Self-Check 8.2.1)

- a) In-Order : 32, 30, 35, 40, 50, 55
- b) Pre-Order : 40, 30, 32, 35, 55, 50
- c) Post-Order : 32, 35, 30, 50, 55, 40

QUESTION 5

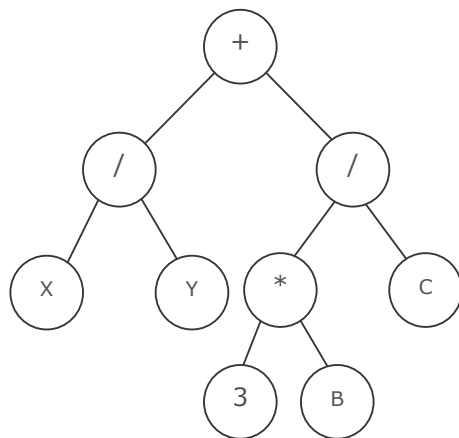
Draw an expression tree corresponding to each of the following:

- a) Inorder traversal is $x / y + 3 * b / c$ (Your tree should represent the Java meaning of the expression.)

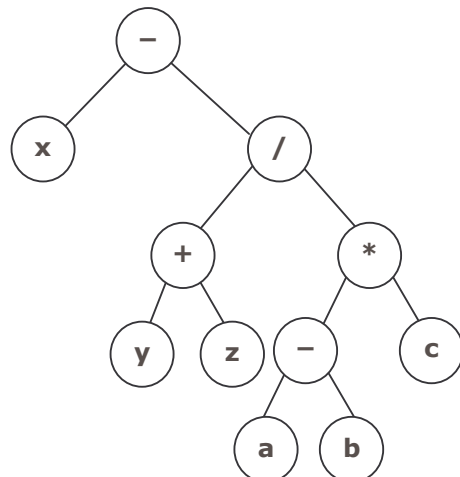
- b) Postorder traversal is $x y z + a b - c * / -$

(Koffman Self-Check 8.2.2)

a)

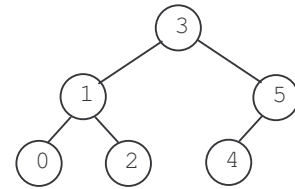


b)



QUESTION 6

Write the string that would be displayed by the `toString` method of the `BinaryTree<E>` class (ref pg 413 Koffman) for the tree shown to the right.
(Koffman Self-Check 8.3.4)



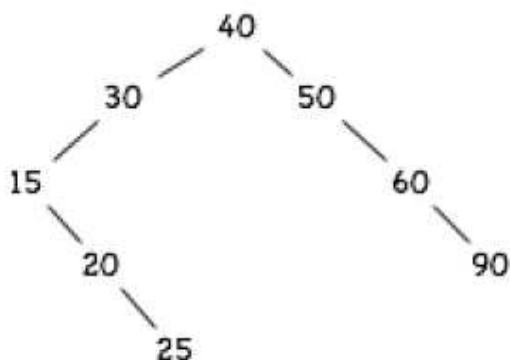
3
 1
 0
 null
 null
 2
 null
 null
 5
 4
 null
 null
 null

QUESTION 7

Show the binary search tree that would be formed by inserting the following data items in order:

45, 30, 15, 50, 60, 20, 25, 90

(Koffman Self-Check 8.4.1)



QUESTION 8

Explain how the tree shown in Figure 8.13 of the Koffman text (pg 417) would be changed if you inserted the word "mother". The word "jane"? Does either of these insertions change the height of the tree?
(Koffman Self-Check 8.4.2)

*mother is added as the right child of morn. jane is added as the left child of kept.
None of these insertions change the height of the tree.*

QUESTION 9

Explain the effect of removing the nodes kept and cow from the tree in Figure 8.13 of the Koffman text (pg 417).

(Koffman Self-Check 8.4.3)

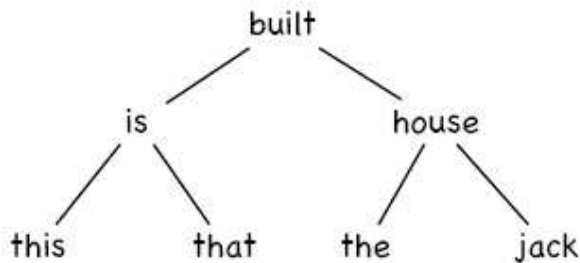
Removing node "kept" will have no other effect, as the node has no children. Node "killed" now has a left child of null. Removal of node "is" will result in the child node "in" moving up into its spot, and that is all. Node "jack" now has the left child of "in". Removal of node "cow" will result in node "cow" being replaced by the largest item in its left sub-tree, namely node "corn". Node "corn" now has a left child of "build" and a right child of "dog". Node "cock" now has a right child of null.

QUESTION 10

Show the heap that would be used to store the words "this", "is", "the", "house", "that", "jack", "built", assuming they are inserted in that sequence. Would exchanging the first and last words change the heap that is formed?

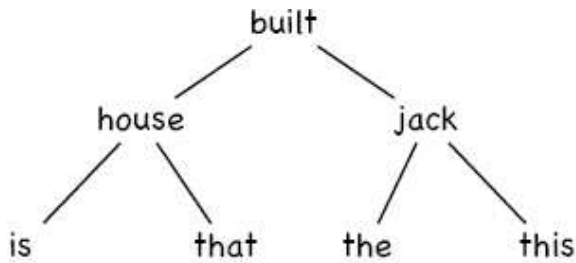
(Koffman Self-Check 8.5.1)

a)



b) first and last words exchanged

yes



QUESTION 11

Draw the heap for Question 10 as an array.

(Koffman Self-Check 8.5.2)

0	1	2	3	4	5	6
built	is	house	this	that	the	jack

QUESTION 12

What is the Huffman code for the letters a, j, k, l, s, t, v using the Huffman tree on page 448 of the Koffman text?

(Koffman Self-Check 8.6.1)

- a 1010
- j 1100001011
- k 11000011
- l 10111
- s 0011
- t 1101
- v 1100000

QUESTION 13

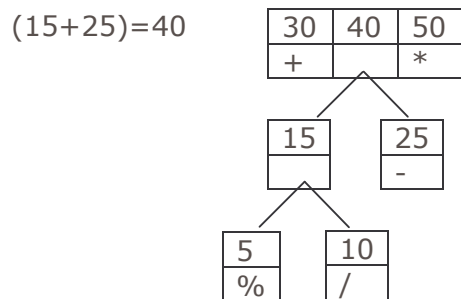
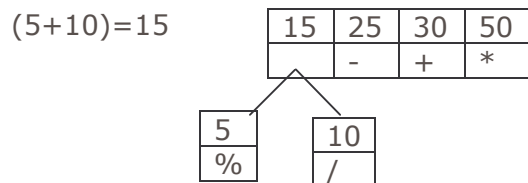
Create the Huffman code tree for the following frequency table:

SYMBOL	FREQUENCY
*	50
+	30
-	25
/	10
%	5

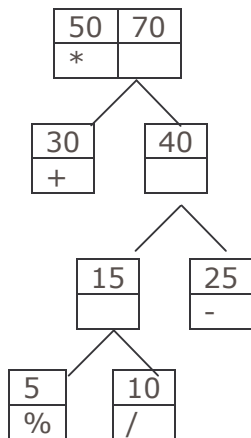
(Koffman Self-Check 8.6.2)

Frequency Table

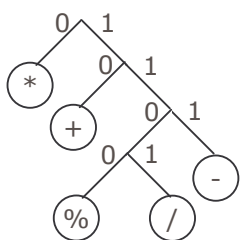
5	10	25	30	50
%	/	-	+	*



$(30+40)=70$



Huffman Tree of *,+,-,/,%



Huffman Code for *,+,-,/,%

SYMBOL	CODE
*	0
+	10
-	111
/	1101
%	1100

CLASS ACTIVITIES

Complete the following activities in groups, asking questions of each other. This will form the basis for class discussion.

ACTIVITY 1 – CROSS CHECK (10 MINUTES)

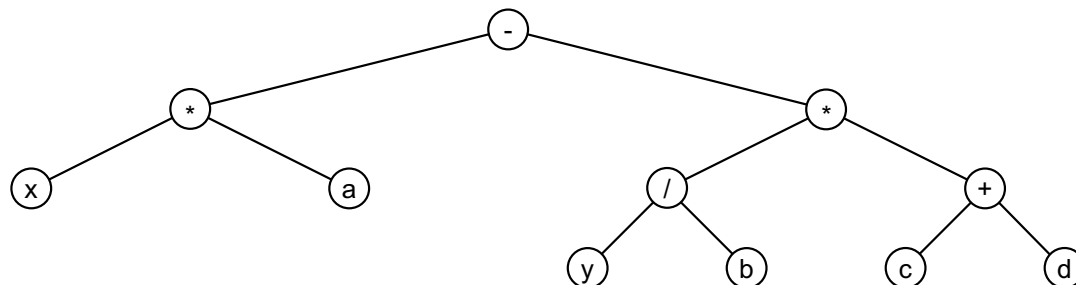
Review each other's solutions to the preliminary tasks, helping each other where necessary. Check that you all agree on your solutions. Ask questions, making sure that you totally understand the material.

ACTIVITY 2 – BINARY EXPRESSION TREE (5 MINUTES)

Draw the binary expression tree for the following infix expression:

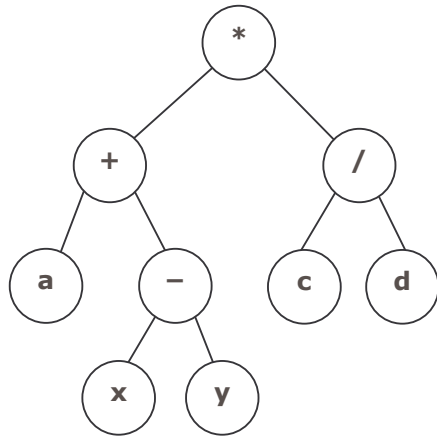
$(x * a) - y / b * (c + d)$

(Koffman Self-Check 8.1.1)



ACTIVITY 3 – PREORDER TRAVERSAL (5 MINUTES)

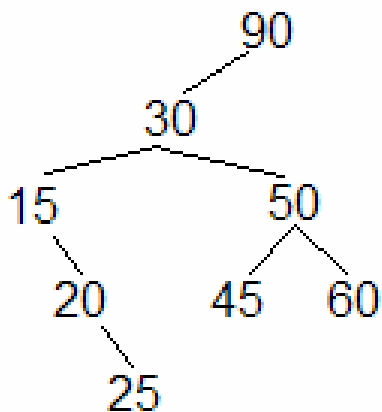
Draw the expression tree that has a preorder traversal given by $* + a - x y / c d$
(Koffman Self-Check 8.2.2)



ACTIVITY 4 – BST (5 MINUTES)

Show the binary search tree that would be formed if the first and last items from the list in question 7 were swapped, but other than that the insertion order remained the same.

(Koffman Self-Check 8.4.1)



ACTIVITY 5 – EQUAL HUFFMAN (5 MINUTES)

What would the Huffman code look like if all symbols in the alphabet had equal frequency?

(Koffman Self-Check 8.6.3)

All of the Huffman codes would be 5 bits in length, except for four of the letters, which would have Huffman codes length 4. The Huffman tree would be a complete binary tree of height 5. The Huffman code would be only slightly more efficient at representing the characters than a normal sequential binary code would.