CSI 2110 Computer Science Fall 2016 University of Ottawa

Assignment #3 (25 points, weight 5%)

Due: Saturday October 22, 11:59PM (accepted without late penalties until Monday October 24, 11:59PM)

The assignment is to be uploaded on blackboard electronically (you may type or write by hand legibly and scan it). Only a single PDF file is accepted.

1. (1 pt) Draw a **binary tree** T that simultaneously satisfies the following:

Each internal node of T stores a single character.

A **preorder** traversal of T yields A B C D E F G H I

An **inorder** traversal of T yields C B E D A H G F I

2. In this question you will sort an array using in-place heapsort (parts a) and b)) and practice insertions on a heap in part c).

a) (4 pt) Build a max-heap using the bottom-up heap construction, in place, in the following array. Show the array after each call to procedure "down-heap":

Index i	0	1	2	3	4	5	6	7	8
A[i]	5	3	17	10	84	19	6	22	9

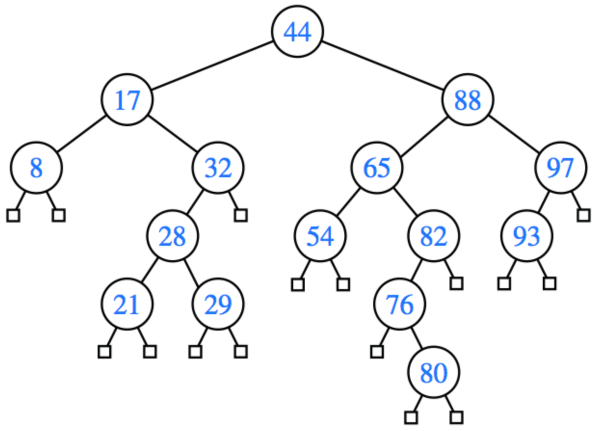
b) (4 pt) After the max-heap is constructed, do the second stage of heapsort, showing the array after each of the 8 remove-max operations, specifying which part of the array is the heap and which part is the sorted sequence.

c) (4 pt) Go back to the original array given in part a). Build a max-heap by successively inserting A[0],A[1],…, A[8] in this order into an initially empty heap stored in an array B. Show B after each of the 9 insertion.

3. (2 pt) Insert the following keys into an initially empty **binary search tree** in the given order and show the final tree:

32, 29, 88, 44, 54, 76, 82

4. Given the following binary search tree:



(a) (2 pt) Show the tree after deleting key 32 and 54.

(b) (2 pt) Using the original tree listed above, show the tree after deleting key 65

5. (2 pt) Give the pseudocode of a recursive method "int calculateHeight(Node p)" that, given the root node of a binary tree, computes the height of the binary tree. For example, in the tree above where the root is the node containing key 44, calculateHeight(root) would return the value 5.

Hint: use some of the tree traversals studied.

Assume as usual that class Node has accessor methods parent(), leftChild() and rightChild() each one returning a Node with obvious meaning.

6. (4 pt) You have learned about full, complete and perfect **binary trees**. This question involves similar concepts but deals with **ternary trees;** a ternary tree is tree where every node has 0, 1, 2 or 3 children. A **full ternary tree** is a ternary tree where every node other than the leaves has 3 children. A **perfect ternary tree** is a full ternary tree with all leaves at the same level. Answer the following questions (short anwers):

(a) (0.5 pt) What is the maximum number of nodes a **ternary tree** can have at level i?

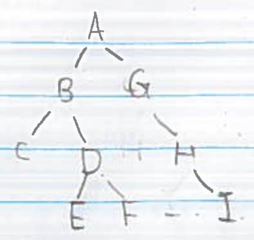
(b) (0.5 pt) What is the number of nodes in a **perfect ternary tree** of height 2?

(c) (1 pt) What is the number of nodes in a **perfect ternary tree** of height h?

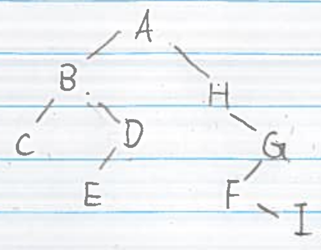
(d) (1 pt) Show a **full ternary tree** of height h=2 with the **minimum** possible number of nodes and a **full ternary tree** of height h=2 with the **maximum** possible number of nodes.

(e) (1 pt) What is the **minimum** and **maximum** number of nodes in a **full ternary tree** of height h?

1. Pre-order from root to leafs

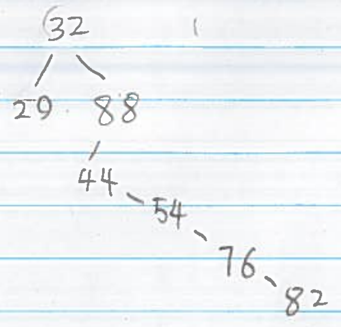


(b). Post order

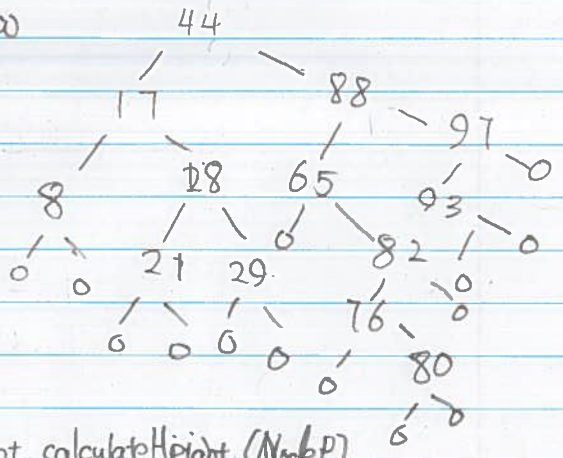


2. [5,] [5,3] [17,3,5] [17,10,5,3] [84,17,5,10,3]
[84, 17, 19, 10, 3, 5] [84, 17, 19, 10, 3, 5, 6]. [84, 22, 19, 17, 3, 5, 6, 10]
[84, 22, 19, 17, 3, 5, 6, 10, 9]

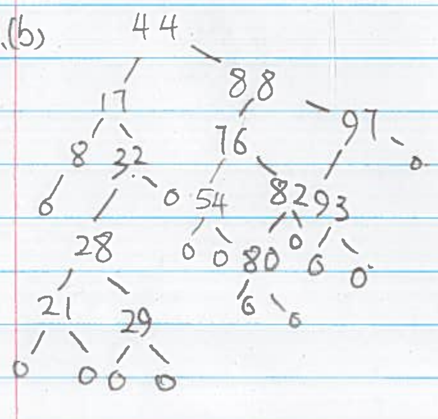
3.



4.(a)

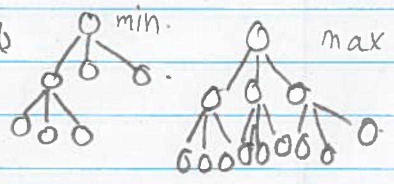


4.(b)



5.
```
int calculateHeight (Node P)
    if P == null
        return 0.
    else  int count = 0.
          calculate Height (P.left  int Count)
          Calculate Height (P.right, int Count)
          return count
```

6. (a) 3^i (b). 13. (c) $\frac{3^{h+1}-1}{2}$ (d)



(e) min $3h+1$
max $\frac{3^{h+1}-1}{2}$.