CS101 Algorithms and Data Structures
Fall 2022
Homework 6

Due date: 23:59, October 30th, 2022

1. Please write your solutions in English.
2. Submit your solutions to gradescope.com.
3. Set your FULL name to your Chinese name and your STUDENT ID correctly in Account Settings.
4. If you want to submit a handwritten version, scan it clearly. `CamScanner` is recommended.
5. When submitting, match your solutions to the problems correctly.
6. No late submission will be accepted.
7. Violations to any of the above may result in zero points.

1. (?? points) Multiple Choices

Each question has **one or more** correct answer(s). Select all the correct answer(s). For each question, you will get 0 points if you select one or more wrong answers, but you will get 1 point if you select a non-empty subset of the correct answers.

Write your answers in the following table.

(a)	(b)	(c)	(d)
C	D	ACD	A

(a) (2') Which of the followings are true?

A. There exists some subtree of a BST such that itself is not a BST.

B. The worst-case of searching an element with specific value in a BST is O(logn).

C. The worst-case of finding the maximum element in a BST is O(n).

D. For a BST, pre-order traversal gives the elements in ascending order.

(b) (2') Suppose we want to use Huffman Coding Algorithm to encode a piece of text made of characters. Which of the following statements are true?

A. Huffman Coding Algorithm will compress the text data with some information loss.

B. The construction of binary Huffman Coding Tree may have time complexity of $O(n)$, where n is the size of the alphabet size of the text.

C. When inserting nodes into the priority queue, the higher the occurrence/frequency, the higher the priority in the queue.

D. The Huffman codes obtained must satisfy prefix-property, that is, no code is a prefix of another code.

(c) (2') Which of the following statements are true for an AVL-tree?

A. Inserting an item can unbalance non-consecutive nodes on the path from the root to the inserted item before the restructuring.

B. Inserting an item can cause at most one node imbalanced before the restructuring.

C. Removing an item in leaf nodes can cause at most one node imbalanced before the restructuring.

D. Only at most one node-restructuring has to be performed after inserting an item.

(d) (2') Consider an AVL tree whose height is h, which of the following are true?

A. This tree contains $\Omega(\alpha^h)$ nodes, where $\alpha = \dfrac{1+\sqrt{5}}{2}$.

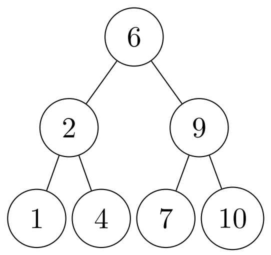
B. This tree contains $\Theta(2^h)$ nodes.

C. This tree contains $O(h)$ nodes in the worst case.

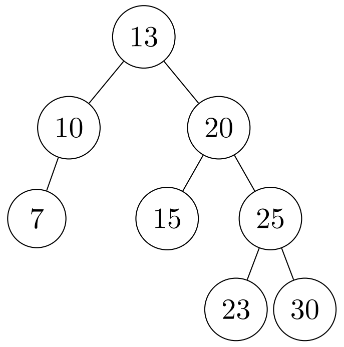
D. None of the above.

2. (?? points) BST and AVL tree

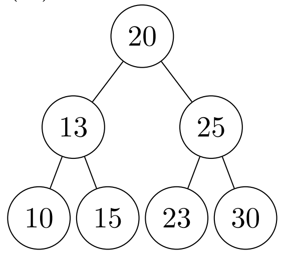
(a) (3') Draw a valid BST of minimum height containing the keys 1, 2, 4, 6, 7, 9, 10.



(b) (4') Given an empty AVL tree, insert the sequence of integers $15, 20, 23, 10, 13, 7, 30, 25$ from left to right into the AVL tree. Draw the final AVL tree.



(c) (2') For the final AVL tree in the question (b), delete 7. Draw the AVL tree after deletion.



(d) (5') For an AVL tree, define D = the number of descendants of the left child of the root - the number of descendants of the right child of the root. Then what is the maximum of D for an AVL tree with height n?

$D_{max} = k_1 \times 2^n + k_2 \times B^n + k_3 \times (-\frac{1}{B})^n$, please write down the value of B and k_i.

since we want to make D_{max}

so we can make the number of descendants of the left child of the root as big as possible, and make the number of descendants of the right child of the root as small as possible.

for the left child, the max situation is that the left subtree is a perfect tree with the height of $n-1$, and since its height is $n-1$,so its number of nodes is $2^n - 1$.

and for right subtree, the min number of nodes to mention the tree as a AVL tree,

so the min height of the right subtree is $n-2$, and the number is min nodes with height of $n-2$ is $F(n-2) = \frac{1}{\sqrt{5}}[(\frac{\sqrt{5}+1}{2})^{n+1} - (\frac{1-\sqrt{5}}{2})^{n+1}] - 1$

so D_{max} =left_max_number-right_min_number= $2^n - 1 - \{\frac{1}{\sqrt{5}}[(\frac{\sqrt{5}+1}{2})^{n+1} - (\frac{1-\sqrt{5}}{2})^{n+1}] - 1\}$

$= 2^n - \frac{1}{\sqrt{5}}(\frac{\sqrt{5}+1}{2})^{n+1} + \frac{1}{\sqrt{5}}(-\frac{1}{\frac{\sqrt{5}+1}{2}})^{n+1}$

$= 2^n - \frac{1+\sqrt{5}}{2\sqrt{5}}(\frac{\sqrt{5}+1}{2})^{n} + \frac{1-\sqrt{5}}{2\sqrt{5}}(-\frac{1}{\frac{\sqrt{5}+1}{2}})^{n}$

$= k_1 \times 2^n + k_2 \times B^n + k_3 \times (-\frac{1}{B})^n$

so above all

$k_1 = 1$

$k_2 = -\frac{1+\sqrt{5}}{2\sqrt{5}}$

$k_3 = \frac{1-\sqrt{5}}{2\sqrt{5}}$

$B = \frac{\sqrt{5}+1}{2}$

3. (?? points) Huffman Coding

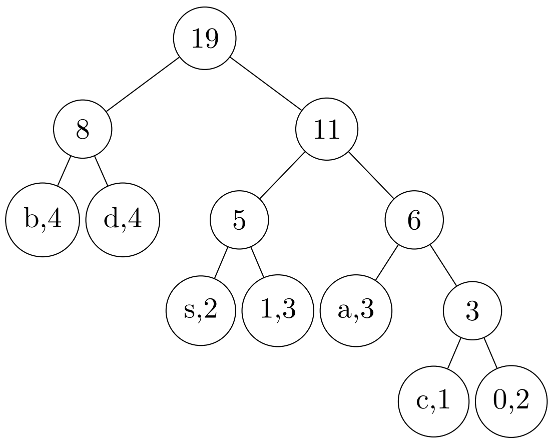
After you compress a text file using Huffman Coding Algorithm, you accidentally spilled some ink on it and you found that one word becomes unrecognizable. Now, you need to recover that word given the following information:

Huffman-Encoded sequence of that word:
11101001011111101
Frequency table that stores the frequency of some characters:

characters	0	1	a	b	c	d	s
frequency	2	3	3	4	1	4	2

(a) (5') Please construct the binary Huffman Coding Tree according to the given frequency table and draw the final tree below.
Note: The initial priority queue is given as below. When popping nodes out of the priority queue, the nodes with the same frequency follows "First In First Out".

c	0	s	1	a	b	d
1	2	2	3	3	4	4



for the inner node: the number is the frequency of all its children,
for the leaf node, the thing before , is the character, the thing after , is the character's frequency.

(b) (3') Now you can "decompress" the encoded sequence and recover the original word you lost. Please write the original word below.

the characters and their huffman code:

b	00
d	01
s	100
1	101
a	110
c	1110
0	1111

so the original word is cs101