University of Waterloo
CS240 Winter 2023
Assignment 3

Due Date: Wednesday, March 8 at 5:00pm

If you are having difficulties with the assignment, course staff are there to help (provided it isn't last minute).

Please read `https://student.cs.uwaterloo.ca/~cs240/w23/assignments.phtml#guidelines` for guidelines on submission. **Each question must be submitted individually to MarkUs as a PDF** with the corresponding file names: a3q1.pdf, a3q2.pdf, ... , a3q6.pdf . It is a good idea to submit questions as you go so you aren't trying to create several PDF files at the last minute.

Late Policy: Assignments are due at 5:00pm, with the grace period until 11:59pm. Assignments submitted after 11:59 on the due date will not be accepted but may be reviewed (by request) for feedback purposes only.

Notes:

- Logarithms are in base 2, if not mentioned otherwise.
- Array indices start from 0.

Question 1 AVL Tree Operations [0+4+3+6=13 marks]

As discussed in class, it is possible to implement AVL trees such that the nodes store only the balance factor $\{-1, 0, 1\}$ at each node instead of the height of the subtree rooted at the node.

a) **Practice** (not worth any marks): Starting from an empty AVL tree, insert the following keys in order. You should obtain the tree shown in figure 1.

$$41, 72, 33, 45, 60, 25, 3, 12, 17, 47$$

b) Show the process of inserting the key 15 into the tree in figure 1. Specifically, draw the tree, with balance factors, after the initial BST `insert` and after each call to `restructure`.

c) Suppose the key 41 were deleted from the *original* tree in figure 1. Would this cause a tree rotation? Briefly justify.

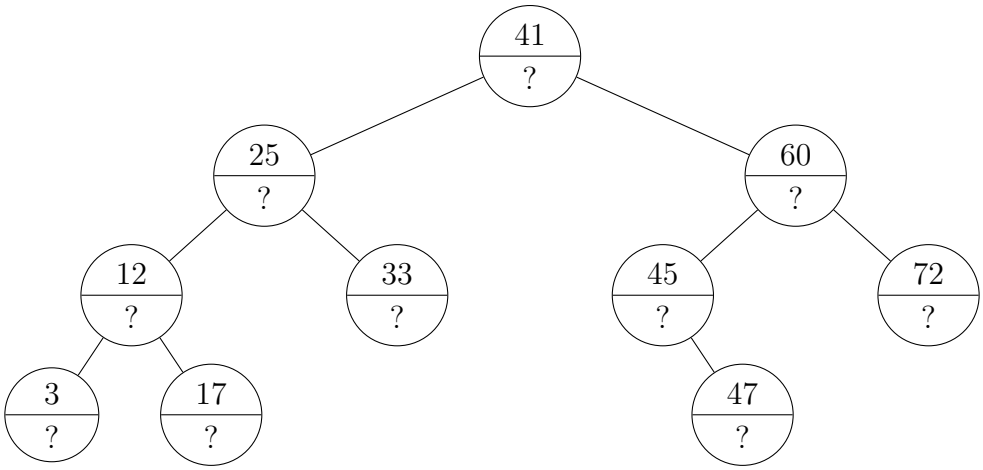

Figure 1: Binary tree T for question 1

d) Show the process of deleting the key 72 from the *original* tree in figure 1. Specifically, draw the tree, with balance factors, after the initial BST `delete` and after each call to `restructure`.

Question 2 AVL Tree Height [3+3+3=9 marks]

In this question, we will prove a recurrence for the height of an AVL tree that leads to a tighter bound (i.e. a lower constant factor) than the proof shown in class.

Let T_h be an AVL tree of height h with a **minimum** number of nodes. Let $N(h)$ be the number of nodes and $N_L(h)$ the number of leaves of T_h.

Recall from class that T_h can be viewed recursively as a tree containing a root with two, possibly empty, subtrees T_{h-1} and T_{h-2}.

a) Prove that for $h \geq 1$, $N(h) = N(h-1) + N_L(h)$.

b) Prove that for $h \geq 0$, $N_L(h) = F(h+1)$, where $F(i)$ is the ith Fibonacci number. ($F(0) = 0$, $F(1) = 1$, $F(n) = F(n-1) + F(n-2)$)

c) Prove that for $h \geq 0$, $N(h) = F(h+3) - 1$. You must use parts a) and b) to prove c).

Question 3 Merging AVL Trees [3+4=7 marks]

a) Describe an algorithm `Join`(T_A, T_B, h_A, h_B) which merges two AVL trees T_A and T_B whose initial heights h_A and h_B are given. You may assume that:

i Every key in A is less than every key in B.

ii h_A and h_B differ by at most one.

Your algorithm must run in time $O(h_A + h_B)$. Justify that your algorithm produces a valid AVL tree and that it runs in the specified runtime.

Erratum: The nodes in T_A and T_B are labeled with heights, not balance factors.

b) Describe an algorithm `Join2`(T_A, T_B, h_A, h_B) as in part a), except that assumption ii is replaced by the following:

iii $h_A < h_B$

As in part a), justify that your algorithm produces a valid AVL tree and that it runs in $O(h_A + h_B) = O(h_B)$ time.

Question 4 Bounded-Height Skip Lists [4+4=8 marks]

Suppose you have a skip list with only three levels. The lower level has $n + 2$ entries $-\infty < a_0 < \cdots < a_{n-1} < +\infty$. The middle one has $k + 2$ entries, where k is an integer that divides n (so that $n = mk$ for some integer m); we assume that (with the exceptions of $\pm\infty$), these entries are evenly spread out so they correspond to $-\infty, a_0, a_m, a_{2m}, \ldots, a_{(k-1)m}, +\infty$. The top level holds $-\infty, +\infty$.

a) What is the worst-case running time for a `search` on this list? Give a $\Theta(\ldots)$ expression involving k and n, and justify your answer.

b) For a given n, explain how to choose k to minimize the worst-case running time. Prove that your choice of k is optimal and give a $\Theta(\ldots)$ bound on the worst-case running time given your choice of k.

Erratum: Your choice of k does not need to divide n.

Question 5 Interpolation Search [3+4+4=11 marks]

a) Execute interpolation search for the key $k = 120$ in the following array:

$[0, 1, 2, 3, 4, 5, 10, 15, 20, 25, 30, 40, 42, 50, 60, 70, 75, 80, 120, 125, 144, 150, 175, 190, 199, 200]$

For each iteration of the loop, your answer must include the values of ℓ and r at the start of the iteration, as well as the value of m, if any, computed in that iteration.

b) Consider interpolation search on a sorted array $A[0, \ldots, n-1]$ that stores real numbers. Suppose that $A[i] = ai + b$ for all $i \in \{0, \ldots, n-1\}$, where a and b are real numbers and $a \neq 0$. For example if $A = \{5, 7, 9, 11, 13, 15\}$ then $a = 2$ and $b = 5$. Prove that the interpolation search on A takes $O(1)$ (no matter if the search is successful or unsuccessful).

c) Recall that the worst-case search time for interpolation search is $\Theta(n)$. Give a family of an arrays A_n, each with n distinct values, (i.e., by defining $A_n[i] = f(i)$ for some function f) as well as search keys k_n such that searching for k_n in A_n has runtime $\Theta(n)$. (There are many examples of such inputs, but you only need one for each n). You must prove the runtime of the algorithm on your inputs is in $\Theta(n)$.

Question 6 Trie Operations [2+2+3+3 = 10 marks]

In this question, when drawing tries, follow the convention that \$ is always to the left of 0, and 0 is always to the left of 1.

a) Draw the standard (uncompressed) binary trie on the following keys:

$$1001, 001, 1111, 10110, 10, 11, 10100, 1, 000, 101, 00$$

Add \$ to the keys as necessary.

b) Repeat part (a), except use a compressed trie.

c) Draw the result of inserting the keys 100 and 1010 into your answer for part (a).

d) Draw the result of deleting the keys 10100, 11, and 1001 from your answer for part (b).