University of Waterloo
CS240 - Winter 2021
Assignment 3
Due Wednesday March 3rd, 5pm.

The integrity of the grade you receive in this course is very important to you and the University of Waterloo. As part of every assessment in this course you must read and sign an Academic Integrity Declaration (AID) before you start working on the assessment and submit it **before the deadline** along with your answers to the assignment; i.e. **read, sign and submit A03-AID.txt now or as soon as possible**. The agreement will indicate what you must do to ensure the integrity of your grade. If you are having difficulties with the assignment, course staff are there to help (provided it isn't last minute).

The Academic Integrity Declaration must be signed and submitted on time or the assessment will not be marked.

Please read `http://www.student.cs.uwaterloo.ca/~cs240/w21/guidelines/guidelines.pdf` for guidelines on submission. **Each written question solution must be submitted individually to MarkUs as a PDF** with the corresponding file names: a3q1.pdf, a3q2.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. **Remember, late assignments will not be marked but can be submitted to MarkUs after the deadline for feedback if you email cs240@uwaterloo.ca and let the ISAs know to look for it.**

Problem 1 [2+6+3=11 marks]

Consider an AVL tree T with n nodes, and let v be a leaf in T. We want to give a lower bound on the *depth* ℓ of v, that is, the length of the path from the root of T to v (the bound will hold for any v).

a) Let $v_0, \dots, v_\ell$ be the path from the root of T to v (where v_0 is the root of T and $v_\ell = v$), and let $T_0, \dots, T_\ell$ be the subtrees of T rooted at respectively $v_0, \dots, v_\ell$. What is T_0 and what is T_ℓ?

b) Prove by induction that for $i = 0, \dots, \ell$, $T_{\ell-i}$ has height at most $2i$.

c) Using the previous question, deduce a lower bound of the form $\ell \in \Omega(g(n))$, for a certain function $g(n)$.

Problem 2 [4+1+3+3+4=15 marks]

Consider the following variant of AVL trees. These are still binary search trees, but we use size instead of height as balancing parameter. Each node has fields `leftDescendants`, `rightDescendants` and `balance`, that contain the number of keys in the left, resp. right, subtrees as well as the balance defined as `balance` = (`rightDescendants`+1) / (`leftDescendants`+1); the terms +1 make sure there is no division by zero, and balances are non-zero.

We say that such a node is *balanced* if the left and right sides are within a multiplicative factor of two of each other, that is, $1/2 \leq$ `balance` ≤ 2 holds. We say that a tree is *balanced* if all its nodes are.

a) Prove that if a node v is balanced, we have `rightDescendants` $\leq (2n-1)/3$, where n is the number of nodes in the tree rooted at v (this includes v itself). Prove a similar inequality for `leftDescendants`.

b) Deduce that `rightDescendants` $\leq \lfloor 2n/3 \rfloor$, and similarly for `leftDescendants`.

c) For $n \geq 1$, let $H(n)$ denote the height of the tallest balanced tree having at most n nodes. Prove that $H(n)$ is a non-decreasing function of n. You will probably have to reuse this question in the rest of the problem.

d) Prove that $H(n)$ satisfies $H(n) \leq 1 + H(\lfloor 2n/3 \rfloor)$ for $n > 1$.

e) Deduce $H(n) \in O(f(n))$, for a certain function $f(n)$ that you have to give us. To do this, consider the smallest integer k such that $n \leq \lceil (3/2)^k \rceil$ and give an upper bound for $H(\lceil (3/2)^k \rceil)$. You can use without proof the fact that for any integer $\ell \geq 1$, $\lfloor 2/3 \lceil (3/2)^\ell \rceil \rfloor \leq \lceil (3/2)^{\ell-1} \rceil$.

Problem 3 [4+4 = 8 marks]

Suppose you have a skip list with only three levels. The lower one 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 $n = km$ 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 time for a query? Give a $\Theta(\ )$ expression involving k and n.

b) Given n, how to choose k to minimize this worst case, and what does the worst case become in that case? Give a $\Theta(\ )$ expression in terms of n.

Problem 4 [2+3+4+5=14 marks]

In class, we saw that skip lists have expected $O(n)$ space requirements and expected height $O(\log(n))$. The proof involves a few steps, which you will generalize for this question. The main difference is that in this problem, we assume that the probability of adding a level to a tower is a fixed number p (with $0 < p < 1$), instead of $\frac{1}{2}$.

You will probably need somewhere the equality $\sum_{i \geq 0} p^i = 1 + p + p^2 + \cdots + p^i + \cdots = 1/(1-p)$.

a) Show that the probability that the tower containing a given key in the skip list has height at least i is p^i. (Here, the height of the tower is the number of times the key is repeated, minus 1.)

b) Assuming there are n distinct keys in the skip list, give the expected number of keys at level i, for $i \geq 0$. Do not count sentinels.

c) Show that the expected number of keys in a skip list is $n/(1-p)$ (again, only count the keys, not sentinel towers.)

d) Show that the expected height of a skip list is at most

$$\log_{1/p}(n) + 1/(1-p).$$

You can assume that n is a power of $1/p$ (for $p = 1/2$, this would mean that n is a power of 2, for instance). In this question, to estimate the height, take into account the left and right sentinels.

Problem 5 [6+6=12 marks]

Consider the list of keys:

$$[1\ 2\ 3\ 4\ 5\ 6\ 7\ 8\ 9\ 10]$$

and assume we perform the following searches in it:

$$10,\ 7,\ 2,\ 2,\ 4^\star,\ 1,\ 2,\ 1,\ 2,\ 1^\star,\ 1,\ 7,\ 1,\ 9^\star$$

1. Using the move-to-front heuristic, give the list ordering after the starred ($\star$) searches are performed.

2. Repeat part (a), using the transpose heuristic instead of the move-to-front heuristic.