University of Waterloo
CS240, Fall 2022
Assignment 4

Due Date: Wednesday, November 16, at 5pm

Please read `https://student.cs.uwaterloo.ca/~cs240/f22/assignments.phtml#guidelines` for guidelines on submission. Submit your solutions electronically as **individual** PDF files with names `a04q1.pdf`, `a04q2.pdf`, `a04q3.pdf` and `a04q4.pdf`, using MarkUs.
There are 49 possible marks plus 5 bonus marks available. The assignment will be marked out of 49.

Problem 1 Hashing [3+3+3+3+2=14 marks]

Consider a hash table dictionary with universe $U = \{0, 1, 2, \ldots, 25\}$ and size $M = 5$. If items with keys $k = 17, 10, 20, 13$ are inserted in that order, draw the resulting hash table if we resolve collisions using:

a) Chaining with $h(k) = (k + 3) \bmod 5$.

b) Linear probing with $h(k) = (k + 3) \bmod 5$

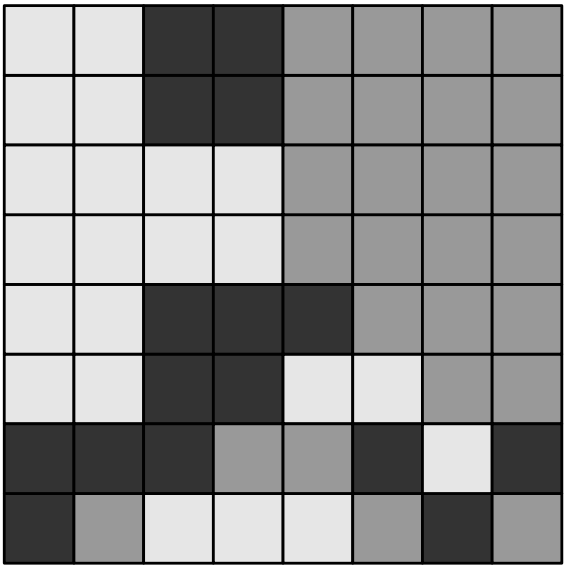
c) Double hashing with $h_1(k) = (k + 3) \bmod 5$ and $h_2(k) = 1 + (k \bmod 4)$.

d) Cuckoo hashing with $h_1(k) = (k + 3) \bmod 5$ and $h_2(k) = \lfloor \frac{k}{5} \rfloor$.

e) Identify a serious problem with the choice of hash functions in part (d).

Problem 2 Quad Trees [6+3+3=12 marks]

a) One of the applications of quad trees is for **image compression**. An image (picture) is recursively divided into quadrants until the entire quadrant is only one colour. Using this rule, draw the quad tree of the following image. Use the convention that each internal node of a quad tree has exactly four children, corresponding to regions R_{NE}, R_{NW}, R_{SW} and R_{SE}, in that order. There are only three colours (shades of grey). For the leaves of the quad tree, use 1 to denote the lightest shade, 2 for the middle shade and 3 for the darkest shade of grey. The bounding boxes in the internal nodes do not need to be indicated.



b) For an integer $k \geq 3$, consider a quad tree with bounding box $[0, 2^k) \times [0, 2^k)$ storing four points $\{(0,0), (2^{k-1}-1, 2^{k-1}-1), (2^{k-1}, 2^{k-1}), (2^k-1, 2^k-1)\}$. Determine the height of the quadtree.

c) For an integer $k \geq 3$, consider a quad tree with bounding box $[0, 2^k) \times [0, 2^k)$ storing four points $\{(0,0), (2^{k-1}, 2^{k-1}), (2^{k-1}+1, 2^{k-1}+1), (2^k-1, 2^k-1)\}$. Determine the height of the quadtree.

Problem 3 Range Trees [5+5+5=15 marks]

a) Assume that we have a set of n numbers (not necessarily integers) and we are interested only in the number of points that lie in a range rather than in reporting all of them. Describe how a 1-dimensional range tree (i.e., a balanced binary search tree) can be modified such that a range counting query can be performed in $O(\log n)$ time (independent of s, the number of nodes). Provide the range counting query and justification of its runtime.

b) Now consider the 2-dimensional-case: We have a set of n 2-dimensional points. Given a query rectangle R, we want to find the number of points that lie in R. Preprocess the n points (by building an appropriate range-tree based data structure) such that you can answer any of these counting queries in time $O((\log n)^2)$. Provide the range counting query and justification of its runtime.

c) Suppose a two dimensional range tree data structure stores n points, and that the x-BST is perfect, i.e., every level is completely filled. Derive an exact closed form formula, in terms of n, for the sum of the number of nodes in the x-BST plus the total number of nodes in all y-BSTs.

Problem 4 Covered Squares [8 marks]

We are given a set $\mathcal{W}$ of n square windows on the computer screen S (i.e., axis-parallel squares in the (x, y)-plane). In addition to a location on the screen, windows also have distinct priorities. If two windows overlap, then the window with greater priority will cover the part of the window with lesser priority that it overlaps with.

a) A new window is created with a priority that differs from all windows in $\mathcal{W}$ and we wish to find all windows in $\mathcal{W}$ that are completely covered by the new window. Give an algorithm that finds all these windows in $O((\log n)^c + s)$ time, where s is the number of windows that are found, and $c \geq 1$ is a constant.

Formally, each window is described via a 4-tuple (x_ℓ, y_b, k, p) with (x_ℓ, y_b) the (x, y)-coordinate of the bottom left corner of the window, k the length of the sides of the window, and p the priority of the window. If the new window is $W' = (x'_\ell, y'_b, k', p')$, your query should return in $O((\log n)^c + s)$ time all those windows $W = (x_\ell, y_b, k, p)$ in $\mathcal{W}$ that are completely covered by the new window.

To make this possible, you will need to assume that $\mathcal{W}$ is stored in a suitable data structure. Describe what data structure you are using. This data structure should take at most $O(n(\log n)^c)$ space, and one should be able to build it in $O(n(\log n)^c)$ time.

b) *5 bonus marks:* For part (a), a correct solution with c any constant will earn full marks. For 5 bonus marks, describe a solution that has $c = 4$.