York University
EECS 4101/5101, Winter 2023
Assignment 4

Due Date: March 19th, at 23:59

Dare to love yourself as if you were a rainbow with gold at both ends ...

Aberjhani

All problems are written problems; submit your solutions electronically **only via Crowdmark**. You are welcome to discuss the general idea of the problems with other students. However, you must write your answers individually and mention your peers (with whom you discussed the problems) in your solution. Please refer to the course webpage for guidelines on academic integrity.

Problem 1 Applications of Hashing [10 marks]

a) Given a set of n positive integers stored in an array A, describe an algorithm that runs in $O(n)$ and outputs any index i such that $A[i]$ and $A[i]^2$ are present in the set. If no such i exists, the algorithm must return -1. Assume indices start at 0. For example, for $A = [1, 8, 2, 3, 49, 2, 5, 7, 10]$, the output could be $i = 7$ (because $A[i] = 7$ and $A[i]^2 = 49$ is also present in the set). You may use a hash table and assume that uniform hashing assumption holds (that is, dictionary operations are supported in constant time).

You can write a pseudocode or describe your algorithm in English. Be succinct; you don't need more than a few sentences to describe a correct algorithm.

b) Given a set of n integers stored in an array A, describe an algorithm that runs in $O(n)$ and outputs the length of the longest sub-array with a sum equal to 0. For example, for $A = [19, -8, -4, 6, -12, 5, 2, 3, 14, 27]$, the longest sub-array with elements summing up to 0 is $[-4, 6, -12, 5, 2, 3]$, and the output must be 6. You may use a hash table and assume that uniform hashing assumption holds (that is, dictionary operations are supported in constant time).

Hint: Form the "prefix-sum" array P, where $P[i]$ is the sum of the first $(i+1)$ indices in A. In the example above, $P = [19, 11, 7, 13, 1, 6, 8, 11, 25, 52]$. You can write a pseudocode or describe your algorithm in English. Be as succinct as you can.

Problem 2 Basics of Hashing [4+4 marks]

Assume that we have a hash table of size $M = 5$, we use the hash function $h(k) = k \bmod 5$, and we use chaining for collision resolution. Furthermore, assume that our universe is $U = \{0, 1, 2, \ldots, 12\}$.

a) Demonstrate the insertion of the keys $0, 1, 2, \cdots, 12$ into the (initially empty) hash table (in that order). You only need to draw the state of the hash table after all insertions are done.

b) [**bonus**] Suppose only two insertions of elements in U are made into the initially-empty hash table. If each pair of elements in U is equally likely to be inserted, calculate the probability that the second insertion caused a collision. In other words, given distinct k_1 and k_2 uniformly chosen from U, find that the probability that $h(k_1) = h(k_2)$.

Problem 3 Hash Functions [5 marks]

Assume a hash scheme in which keys are selected uniformly at random from the Universe set $U = \{1, 2, 3, \ldots, 600\}$. Consider the following two hash functions: $h_1(k) = k \; mod \; 6$ and $h_2(k) = 3k \; mod \; 6$. Which hash function is better? Justify your answer.

Problem 4 Collision Handling [5+5=10 marks]

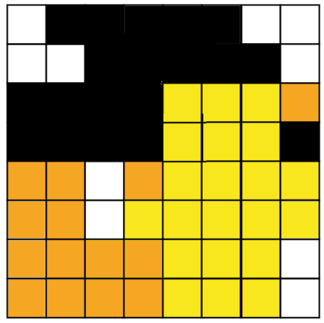
Consider a hash table dictionary with the universe $U = \{0, 1, 2, \ldots, 24\}$ and size $M = 5$. If items with keys $k = 21, 3, 16, 1$ are inserted in that order, draw the resulting hash table if we resolve collisions using:

a) Linear probing with $h(k) = (k + 1) \bmod 5$

b) Cuckoo hashing with $h_1(k) = k \bmod 5$ and $h_2(k) = \lfloor k/5 \rfloor$

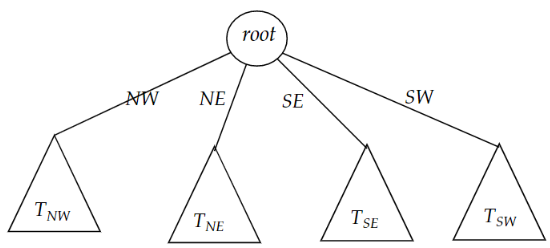
Problem 5 Image Encoding [5+5=10 marks]

Consider the following 8×8 pixel image of a (deformed) pokémon. Each pixel is either white (0), yellow (1), orange (2), or black (3). In this problem, we consider possible ways to encode the image.



a) Consider a fixed-length encoding of the image as follows. We process the image row by row (from top to bottom), and for each row, we process pixels from left to right. For each pixel, we store 2 bits to specify its colour (00 for white, 01 for yellow, 10 for orange, and 11 for black). The first six bits will then be 001111.
Indicate what the length of the code will be. Show your work (you don't need to show the actual code).

b) One application of quadtrees is in image compression. An image is recursively divided into quadrants until the entire quadrant is only one colour. Using this rule, draw the quadtree of the pokémon image. Note that each tree leaf must store the colour of the area it represents. Draw a separate tree for each of $T_{NW}, T_{NE}, T_{SE}, T_{SW}$ in the following figure.



Problem 6 Quad Trees [6 marks]

Let n be a large positive integer. Suppose we create n random numbers uniformly distributed in the range $[0, 1]$ and store them in an array $X = [x_1, x_2, \ldots, x_n]$. Similarly, we store n uniformly-distributed random numbers in $[0..1]$ and store them in an array $Y = [y_1, y_2, \ldots, y_n]$ and store another n uniformly-distributed random numbers (from the same range $[0..1]$) and store them in an array $Z = [z_1, z_2, \ldots, z_n]$. Consider a set $P = \{p_1, p_2, \ldots, p_n\}$ of n 3D points, where $p_i = \{x_i, y_i, z_i\}$. That is, P is a set of points distributed uniformly at random in a unit 3D box. We store P using a quadtree. Specify what the expected height of P is. You need to provide an exact formula and show your work.

Problem 7 KD trees [4+4=8 marks]

Consider the following set of points in two dimensions:
$S = \{(13,14), (11,2), (4,6), (12,10), (9,20), (10,12), (3,14), (14,8), (15,18), (4,4), (6,7), (7,0)\}$.

a) Draw the analogous plane partition diagram and kd-tree.

b) Show how a search for the points in the query rectangle $R = [11.5, 25] \times [14.5, 11]$ would proceed (that is, the box is formed by points (x, y) where $x \in [11.5, 25]$ and $y \in [11, 14.5]$). It suffices to show which nodes in the kd-tree are examined in the search.