University of Waterloo
CS240 Spring 2021
Assignment 4

Written Questions Due: Wednesday, July 14 at 5:00pm

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 before you start working on the assessment and submit it **before the deadline of July 14th** along with your answers to the assignment; i.e. **read, sign and submit A04-AcInDe.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/s21/guidelines.pdf` for guidelines on submission. **Each question must be submitted individually to MarkUs as a PDF** with the corresponding file names: a4q1.pdf, a4q2.pdf, ... , a4q6.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.
Late assignments, however, can be reviewed for **feedback only** upon request to the ISAs at cs240@uwaterloo.ca.
Note: you may assume all logarithms are base 2 logarithms: $\log = \log_2$.

Problem 1 [4+4+4=12 marks]

a) Consider the Interpolation search on a sorted array $A[0, .., n-1]$ 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 $a > 0$. For example if $A = \{5, 7, 9, 11, 13, 15\}$ then $a = 2$ and $b = 5$. Prove that the interpolation search on A always takes $O(1)$ (no matter if the search is successful or unsuccessful).

b) Recall that the worst-case search time for interpolation search is $\Theta(n)$. Give an example of an array A with n distinct values (i.e., by defining $A[i] = f(i)$ for some function f) as well as a search key k that demonstrates the worst-case time; you need to justify the runtime. There are infinitely many such examples, but any one of them will suffice.

c) Suppose $A[i] = t\sqrt{i}$ for all $i \in \{0, \ldots, n-1\}$ and some positive number t. Show that the runtime to search for t using interpolation search is $O(\log \log n)$.

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

a) Draw the standard trie that is obtained after inserting the following keys into an initially empty trie:

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

b) From your answer to part (a), draw the trie that is obtained after removing the following keys:

$$10100\$, 11\$, 1001\$$$

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

d) Repeat part (b), but on the compressed tree that is obtained in (c).

e) Find the exact height of the trie with keys that are binary representations of numbers $0, 1, 2, 3, 4, \ldots, 2^k - 1$ without unnecessary leading 0s, and inserted into the trie in increasing order. That is, insert $0\$, 1\$, 10\$, 11\$, 100\$$, etc.

f) Repeat part (e) using compressed tries. Also, for this part, prove your result, including any structural properties of the compressed tries.

Note: In case you decided to use mathematical induction, solutions must clearly state:

- what you are doing induction on;
- the base case(s);
- the inductive hypothesis;
- the inductive step.

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

a) What is the worst-case running time for inserting n items into an initially empty *hash table* of size *M*, where collisions are resolved by *chaining*? Would it make a difference if you inserted new nodes at the end of the chain rather than at the start (or vice-versa)? Assume that n is greater than M, where M is the number of cells in the hash table. Also assume that inserting at the end of the list is $O(1)$.

b) What would be the worst-case running time if each sequence (each hash chain) was stored in sorted order?

c) What would be the worst-case running time if we stored the root of a binary search tree, *BST*, (instead of the start of a linked list) in the hash table? In other words, we insert new nodes into the BST tree rooted at hash cell location *h(k)*, where k is the key and h is the hash function.

Problem 4 [5+2=7 marks]

a) Suppose that we are using a hash table with M buckets, where $M > 1$. Recall that in open addressing we store an element in the first empty bucket among $h(k, 0), h(k, 1), h(k, 2), \ldots,$ that is empty.
Another version of probing (named quadratic probing) uses the function
$h(k, i) = (h(k) + i^2 + i) \mod M$ where $h(k)$ is some hash function.
Prove that the probe sequence of this method will always include at most $\frac{(M+1)}{2}$ distinct buckets. Explain why is this a problem?

b) Suppose we have a hash table of size M using double hashing, with independent hash functions h_0 and h_1, and probe sequence $h(k, i) = (h_0(k) + i * h_1(k)) \mod M$. Show by contradiction that if M is a prime and $0 < h_1(k) < M$, then double hashing probing sequence for key k visits all slots of the table during the first M attempts.

Problem 5 [6 marks]

Company *TradeSuccess* operates in the stock-market, entering a huge number of financial transactions – called trades – every day. On the other side of each trade, there is some company *Y*, call it a counter-party of *TradeSuccess*. To each trade of *TradeSuccess* is assigned a portfolio number, which is not a unique identifier of the trade, and is not even unique per counter-party. In other words, trades with different counter-parties may be assigned the same portfolio, and different trades with the same counter-party may be assigned distinct portfolios as well, according to some internal policy of the company.
Whenever *TradeSuccess* enters into a trade, a new line such as "+ 1005 26" is appended to a log file, where "+" indicates that a new trade was initiated by *TradeSuccess*, "1005" is the counter-party id, and "26" is the portfolio of that trade. The log file will not add in a trade

whose counter-party and portfolio are identical to a currently active trade.
At any time during a typical day, however, a counter-party Y may withdraw all transactions with *TradeSuccess*. From *TradeSuccess*'s standpoint, that means all trades it had entered into with Y that day up to this point in time are now considered void, and a line such as "-101" is logged. Here, the "-" sign indicates that a cancellation took place, and "101" is the id of the counter-party the cancellation refers to. A cancellation can only occur when there is at least one active trade with the cancelling counter-party.
The following sample log file illustrates the basic structure of the log file of *TradeSuccess*.

```
+  1005 26
+  2005 25
+  1005 25
+  5555 666
-  1005
+  5555 26
+  2222 10
+  1005 13
-  2005
```

`You may assume that there are no duplicated lines in the file.`

By the end of each day, the company *TradeSuccess* needs to determine the set of distinct portfolios associated to trades that are still active by then, that is, the portfolios of trades that have not been canceled during the day.
Write an algorithm that takes a log file for a single day and returns the set of distinct portfolios associated to trades that are still active in expected running time or average-case running time of $O(n)$ where n in the length of the log file.
Note: applying the algorithm on the log file above should return the set 666,26,10,13.

Problem 6 [4+6+2=12 marks]

a) Give the (x, y) coordinates of three points in the plane for which the corresponding quadtree has height exactly 6. (Do not give the plane partitions.)

b) The *min-max spread* of a point set is the ratio of the maximum distance between points to the minimum distance between points. Prove that spread of any set of n points in the plane is $\Omega(\sqrt{n})$.
Hint: Define M to be the maximum distance and m to be the minimum distance. You need to prove that min-max spread $= \frac{M}{m} \in \Omega(\sqrt{n})$.
Consider defining a shape that includes any set of n points in the plane. Then consider how mathematically you can write a correct expression/fact/observation that can be simplified to conclude that min-max spread $= \frac{M}{m} \in \Omega(\sqrt{n})$.

c) A set of n points in the plane is called *dense* if its *min-max spread* is $O(\sqrt{n})$. Prove that the height of the quadtree on any dense point set is $O(\log n)$.
Hint: You might want to consider finding an upper bound R in terms of M.