University of Waterloo
CS240E, Spring 2026
Written Assignment 4

Due Date: Tuesday, July 14, 2026 at 5:00pm

Be sure to read the assignment guideliness (`https://student.cs.uwaterloo.ca/~cs240e/s26/assignments.phtml#guidelines`). Submit your solutions electronically to Crowdmark. Ensure you have read, signed, and submitted the **Academic Integrity Declaration** AID02.

Grace period: submissions made before 7:59pm on July 14 will be accepted without penalty. Please note that submissions made after 7:59pm **will not be graded** and may only be reviewed for feedback.

Question 1 [2+4=6 marks]

Suppose that we use hashing with a randomly picked hash-function and the uniform hashing assumption holds. As usual, let M be the table-size, and let $n \leq M$ be the number of key-value pairs that we want to insert.

For both part-questions, give an exact answer (no asymptotics), simplify your expression as much as possible, and justify your answer.

a) What is the probability that the first and second item that we inserted are in distinct buckets?

b) What is the probability that after inserting n items there are no collisions, i.e., all buckets contain at most one item?

Hint: The formula that you give should depend on both n and M. It also should be 1 for $n = 1$, and the same as the answer to (a) for $n = 2$. Be sure to check this!

Question 2 [1+2+2+5=10 marks]

Assume that we have a hash function h, and define a probe sequence via $h(k, 0) = h(k)$ and

$$h(k, i) = \Big(h(k, i{-}1) + i\Big) \bmod M \quad \text{for } 1 \leq i < M$$

a) Write the probe sequence for $h(k) = 0$ and $M = 8$.

b) A probe sequence is called *quadratic probing* if $h(k, i) = \big(h(k) + c_1 i + c_2 i^2\big) \bmod M$ for some hash-function h and some constants $c_1, c_2 \geq 0$. Show that the probe sequence defined above is an instance of quadratic probing.

c) Show that if $h(k, i){=}h(k, j)$ for some $0 \leq i < j < M$, then $(j{-}i)(j{+}i{+}1){=}0 \bmod 2M$.

d) Assume that M is a power of 2, say $M = 2^m$ for some integer m. Prove that all entries in the probe sequence are different.

Hint: The course notes contains some rules about modular arithmetic that you may use without proof.

Question 3 [2+4+5 = 11 marks]

We have seen one method of obtaining a universal family of hash-functions in class. This assignment discusses another one. Let us assume that all keys come from some universe $\{0, \ldots, U-1\}$, where $U = 2^u$. Therefore any key k can be viewed as bit-string x_k of length u by taking its base-2 representation.

Let us assume further that the hash-table-size M is $M = 2^m$ for some integer m, with $m < u$. To choose a hash-function, we now randomly choose each entry in a $m \times u$-matrix H to be 0 or 1 (equally likely). Then compute $h_k = (Hx_k)\%2$, where x_k is now viewed as a vector and '%2' is applied to each entry. The output is a m-dimensional vector with entries in $\{0, 1\}$; interpreting it as a length-m bit-string gives a number $\{0, ..., M-1\}$ that we use as hash-value $h(k)$. For example, if $k = 18$, $u = 5$, $m = 3$ and H is as shown below, then $h(k) = 1$ since

$$\underbrace{\begin{pmatrix} 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix}}_{H} \quad \underbrace{\begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 0 \end{pmatrix}}_{\text{18 as length-5 bit-string}} \quad \%2 = \underbrace{\begin{pmatrix} 0 \\ 2 \\ 1 \end{pmatrix}}_{Hx_k} \%2 = \quad \underbrace{\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}}_{\text{1 as length-3 bit-string}}$$

a) Let H be the above matrix, $u = 5$ and $m = 3$. Consider the keys 9 and 13. What are their hash-values? Show your work.

b) Consider again $u = 5, m = 3$ and keys $k = 9$ and $k' = 13$. Consider the same matrix H, except that the bits in the middle column are randomly chosen. What is the probability that $h(k) = h(k')$? Justify your answer.

c) Show that (for any u, m) this method of choosing the hash function gives a universal hash function family, or in other words, $P\big(h(k) = h(k')\big) \leq \frac{1}{M}$ for any two keys $k \neq k'$.

Question 4 [5+5+2=12 marks]

A *range-counting-query* is like a range search, except that you only need to report *how many* items fall into the range, you do not need to list which items they are.

a) Describe how any 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 points in the query-interval). Briefly state the changes needed, then describe the algorithm for the range counting query.

b) Now consider the 2-dimensional-case: Describe an appropriate range-tree based data structure such that you can answer range-counting-queries among 2-dimensional points in time $O((\log n)^2)$. Then describe the algorithm for the range counting query.

c) Assume now that the range-counting query is 3-sided. Which data structure for storing points would you use if your objective is a small run-time? Briefly (in 2-3 sentences) justify your answer.

Question 5 Bonus [(+5) marks]

Assume you are given an array P of n points, where points are in general position and sorted by x-coordinates. Describe an algorithm that builds a priority search tree to store points P, and that has $O(n)$ *worst-case* time (partial credit will be given for $O(n)$ expected time).

Note: the textbook suggests to use the median x-coordinate as split-line coordinate, but you are allowed to use any values for the split-line coordinates as long as the resulting priority search tree has height at most $\lceil \log n \rceil$.