Computer Science CSC263H
St. George Campus

February 13, 2020
University of Toronto

Homework Assignment #4

Due: February 27, 2020, by 5:30 pm

- **You must submit your assignment through the Crowdmark system.** You will receive by email an invitation through which you can submit your work. If you haven't used Crowdmark before, give yourself plenty of time to figure it out!
- You must submit a **separate** PDF document with for **each** question of the assignment.
- To work with one or two partners, you and your partner(s) must form a **group** on Crowdmark (one submission only per group). We allow groups of up to three students, submissions by groups of more than three students will not be graded.
- The PDF file that you submit for each question must be typeset (**not** handwritten) and clearly legible. To this end, we encourage you to learn and use the LaTeX typesetting system, which is designed to produce high-quality documents that contain mathematical notation. You can use other typesetting systems if you prefer, but handwritten documents are not accepted.
- If this assignment is submitted by a group of two or three students, for each assignment question the PDF file that you submit should contain:
  1. The name(s) of the student(s) who *wrote* the solution to this question, and
  2. The name(s) of the student(s) who *read* this solution to verify its clarity and correctness.
- By virtue of submitting this assignment you (and your partners, if you have any) acknowledge that you are aware of the homework collaboration policy that is stated in the csc263 course web page: `http://www.cs.toronto.edu/~sam/teaching/263/#HomeworkCollaboration`.
- For any question, you may use data structures and algorithms previously described in class, or in prerequisites of this course, without describing them. You may also use any result that we covered in class, or is in the assigned sections of the official course textbook, by referring to it.
- Unless we explicitly state otherwise, you should justify your answers. Your paper will be marked based on the correctness and completeness of your answers, and the clarity, precision, and conciseness of your presentation.
- The total length of your pdf submission should be no more than 4 pages long in a 10pt font.

Question 1. (1 marks)

A famous csc263 alumni, Tony Soprano, proposes the following method of hashing that reduces the number of collisions when searching in a hash table. Suppose we have a hash table T with m slots and two ***independent*** hash functions h_1 and h_2. Each hash function satisfies the ***Simple Uniform Hashing Assumption*** (SUHA): every key in the universe U hashes with equal probability to any hash slot of T, independently from the hash values of all other keys in U. When we insert a new element x, we add it to the chain in one of the two hash slots $T[h_1(x)]$ or $T[h_2(x)]$; we pick the slot in which the chain is shorter. If the chains are same length, we add x to the chain in $T[h_1(x)]$.

a. If the number of ***non-empty*** slots in T is k before inserting x, what is the probability that x is inserted into an ***empty*** slot? Justify your answer.

b. Suppose that $m = 4$, and $T[0]$ contains 4 elements, $T[1]$ contains 2 elements, and $T[2]$, $T[3]$ contain 6 elements each. What is the ***expected*** length of the chain x is inserted into, ***not counting*** x itself? Justify your answer.

Question 2. (1 marks)

A large number n of dinosaur bones are discovered at an archeological site, and each bone is labeled by a ***distinct*** integer in $\{1, 2, \ldots, n\}$. Researchers examine some pairs of these bones to try to determine the number of different ***species*** of dinosaurs the bones come from. From this examination, they create a list L that contain a total of m items: each item in L is of the form $S(i, j)$, which means the bone labelled i and the bone labelled j are from the ***same*** specie, or of the form $D(k, l)$, which means that the bone labelled k and the bone labelled l are from ***different*** species. But, unfortunately, researchers may make mistakes: for example, they could create a list $L = \ldots, S(3, 2), \ldots, D(5, 2), \ldots, S(5, 3), \ldots$ which must be incorrect (do you see why?).

Design an efficient algorithm for the following task:

- The algorithm's input is a list L of m items as described above, where $m \geq n$.

  (Recall that n is the total number of bones that were discovered.)

- ***If*** the list L is incorrect (as explained above)

  then the algorithm outputs ERROR FOUND

  else the algorithm outputs an integer k, where k is the maximum possible number of different species the bones are from, according to the input list L.

Describe your algorithm in ***clear and concise*** English and analyze its worst-case time complexity. ***The worst-case running time of your algorithm must be asymptotically better than*** $O(mn)$***.***

HINT: Use a data structure that we learned in class.

[***The questions below will not be corrected/graded. They are given here as interesting problems that use material that you learned in class.***]

Question 3. (0 marks) Consider the sorting algorithm given by the pseudocode below. It takes an array $A[1\,..\,n]$ of size n, and outputs A with its elements in sorted (non-decreasing) order.

```
1  for i = 2 to n
2      j = i − 1
3      while A[j + 1] < A[j]  &  j ≥ 1
4          swap A[j] and A[j + 1]
5          j = j − 1
```

In the following subquestions, assume that the array A contains a uniformly chosen random permutations of the integers $1, \ldots, n$.

a. Let S_i be the number of swaps performed by the algorithm in the i-th iteration of the for-loop. What is the ***exact expected value*** of S_i as a function of n and i? Justify your answer.

b. Let $S = S_1 + \ldots + S_{n-1}$ be the total number of swaps performed by the algorithm. What is ***exact expected value*** of S as a function of n? Justify your answer.

Question 4. (0 marks) Consider a singly linked list containing the numbers $0, 1, \ldots, n$ ($n \geq 1$). The "Move To Front" operation $MTF(i)$ searches the list from the beginning until i is found and then moves i to the beginning of the linked list.

The list is initially sorted in increasing order, and a sequence of $m \geq 1$ operations $MTF(i_1), \ldots, MTF(i_m)$ is performed, where $i_1, \ldots, i_m$ are chosen independently from $\{0, 1, \ldots, n\}$, with $\text{Prob}[i_j = n] = \frac{1}{2}$ and $\text{Prob}[i_j = \ell] = \frac{1}{2n}$ for $\ell \neq n$ (for each j, $1 \leq j \leq m$).

Let L be the list after $MTF(i_1), \ldots, MTF(i_m)$ are performed.

a. Describe the probability space for this problem: what is the sample space, and what is the probability distribution (i.e., the probability of each event in the sample space)?

b. Let $\ell \in \{0, 1, \ldots, n\}$. What is the probability that $\ell \notin \{i_1, \ldots, i_m\}$? Briefly explain why.

c. For $\ell \in \{0, 1, \ldots, n-1\}$, let P_ℓ denote the probability that n appears before ℓ in L. Compute the value of P_ℓ. Simplify your expression. Briefly justify your answer.

d. For $\ell \in \{0, 1, \ldots, n-1\}$, let B_ℓ be the indicator variable that is 1 if ℓ appears before n in L and 0 otherwise. What is the expected value $E[B_\ell]$of B_ℓ? Simplify your expression.

e. Let T be the number of list elements examined if $MTF(n)$ is performed on the list L. What is the expected value $E[T]$ of T as a function of $E[B_\ell]$?
What is the expected value $E[T]$ of T as a function of n and m? Simplify your expression.

NOTE: To partially verify your answers above, consider the special case that $m = 1$ and check that your answers are correct for this simpler case.

Question 5. (0 marks)

We are given as input the set of vertices $V = \{1, \ldots, n\}$ and a sequence of edges $e_1, \ldots, e_m$, where for each $1 \leq i \leq m$, $e_i = (j, k)$ for some $j, k \in V$, $j \neq k$. For each $0 \leq i \leq m$, define the undirected graph $G_i = (V, E_i)$ where $E_0 = \emptyset$, and for $i \geq 1$, $E_i = \{e_1, \ldots, e_i\}$. Design an algorithm that outputs an array $C[0..m]$, with $C[i]$ equal to the number of nodes of the largest connected component of G_i (note that $C[0] = 1$). Your algorithm should run in time $O(n + m \log^*(n))$.

Describe the algorithm in clear and concise English. Explain why it is correct and why it runs in the required time complexity.

Hint: See "An application of disjoint-set data structures" in pages 562-564 of CLRS (Chapter 21).