Foundations of Algorithms, Fall 2022
Homework #5

In this course, critical thinking and problem analysis results in discovering appropriate and better, the best, algorithm for a problem; defining the algorithm in pseudocode; demonstrating (we don't usually prove) that the algorithm is correct; and determining the asymptotic runtime for the algorithm using one or more of the tools for asymptotic analysis you have studied this semester. Each algorithm you design must follow the **Pseudocode Restrictions** (Canvas page *Pseudocode Restrictions & Programming Guidelines*) when preparing your solutions to these problems.

Exercising the algorithm you design using one or more example data set(s) removes any doubt from the grader that the algorithm is structurally correct and all computations are correct. When a problem supplies data you are expected to use it to demonstrate that your algorithm is correct.

All members of the collaboration group are expected to participate fully in solving collaborative problems, and peers will assess performance at the end of the assignment. Note, however, that each student is required to write up their solutions individually. Common solution descriptions from a collaboration group will not be accepted. Furthermore, to receive credit for a collaboration problem, each student in the collaboration group must actively and substantially contribute to the collaboration. This implies that no single student should post a complete solution to any problem at the beginning of the collaboration process.

You are permitted to use Internet resources while solving problems, however complete references must be provided. Please follow the Sheridan Libraries' citation guidance at "*Citing Other Things* - HOW DO YOU CITE AN INTERVIEW, A TWEET, OR A PUBLIC WEB PAGE?" and continue to the APA Academic Writer *Sample References*. Additional example citations are provided at the Purdue University, Purdue Online Writing Lab (OWL), College of Liberal Arts *Reference List: Electronic Sources*. You can also use training provided by Victoria University Library, Melbourne Australia on *Harvard Referencing: Internet/websites*

All written assignments are expected to be typed in a word processor, the preferred submission type is PDF allowing for ease of grading. When using equations, make use of the built-in equation editor for these tools. Try to avoid scanning hand-drawn figures, if needed please ensure they are readable. Only scans from flat bed scanners or printers will be accepted; all scans from phones or phone photos will be rejected. The grader will make the determination of "readability". If the assignments are not legible, they will be returned to the student with a grade of zero.

1. [20 points] CLRS 16.3-6: Design a data structure to support the two following operations for a dynamic multiset S of integers, which allows duplicate values:

   (a) INSERT(S, x) inserts x into S,

   (b) DELETE-LARGER-HALF(S) deletes the largest $\lceil |S|/2 \rceil$

   Explain how to implement this data structure so that any sequence of m INSERT and DELETE-LARGER-HALF operations run in $O(m)$ time. Your implementation should also include a way to output the elements of S in $O(|S|)$ time.

2. [20 points] CLRS 19.3-3: Give a sequence of m MAKE-SET, UNION, and FIND-SET operations, n of which are MAKE-SET operations that take $\Omega(mlgn)$ time when using only union by rank and not path compression.

3. [30 points] CLRS 19.4-2 and 19.4-3:

   (a) [15 points] Prove that every node [in a disjoint-set forest] has rank at most $\lfloor lgx \rfloor$

   (b) [15 points] In light of Exercise 19.4-2, how many bits are necessary to store $x.rank$ for each node in x?

   (Continued on next page)

4. [30 points] ***Collaborative Problem*** - In this problem we consider two stacks A and B manipulated using the following operations (n denotes the size of A and m the size of B):

- PushA(x): Push element x on stack A.
- PushB(x): Push element x on stack B.
- MultiPopA(k): Pop $min(k, n)$ elements from A.
- MultiPopB(k): Pop $min(k, m)$ elements from B.
- Transfer(k): Repeatedly pop an element from A and push it on B, until either k elements have been moved or A is empty.

Assume that A and B are implemented using doubly-linked lists so that PushA and PushB, and single pop from A or B, can be performed in $O(1)$ time worst-case.

(a) [5 points] What is the worst-case running time of the operations MultiPopA, MultiPopB, and Transfer?

(b) [15 points] Define a potential function $\Phi(n, m)$ for all of the operations (PushA(x):, PushB(x), MultiPopA(k), MultiPopB(k), and Transfer(k):). **You must explain how you determined your potential function. Stating a potential function without this explanation will earn 0 of 15 points on this part of the problem**.

(c) [10 points] Use your potential function $\Phi(n, m)$ to prove that the operations $MultiPopA$, $MultiPopB$, and $Transfer$ have amortized running time $O(1)$.