CS 430 – FALL 2023
INTRODUCTION TO ALGORITHMS
HOMEWORK #6

1. (6 points) Assume you are creating an array data structure that has a fixed size of n. You want to backup this array after every so many insertion/update operations. Unfortunately, the backup operation is quite expensive, it takes n time to do the backup, regardless of how many items are currently in the data structure. Insertions/updates without a backup just take 1 time unit.

   1a) How frequently can you do a backup and still guarantee that the amortized cost of insertion/update is O(1)?

   1b) Prove that you can do backups in O(1) amortized time.

2. (7 points) Suppose we wish not only to increment a counter but also to reset it to zero (i.e., make all bits in it 0). Counting the time to examine or modify a bit as $\Theta(1)$, show how to implement a counter as an array of bits so that any sequence of n INCREMENT and RESET operations takes time $O(n)$ on an initially zero counter. You must use amortized analysis. (Hint: Keep a pointer to the high-order 1.)

3. (7 points) **Rooted Fibonacci trees** T_n are defined recursively in the following way. T_1 and T_2 are both the rooted tree consisting of a single vertex, and for $n = 3, 4, \ldots,$ the rooted tree T_n is constructed from a root with T_{n-1} as its left subtree and T_{n-2} as its right subtree.

   3a) Draw the first seven rooted Fibonacci trees.

   3b) How many vertices, leaves, and internal vertices does the rooted Fibonacci tree T_n have, where n is a positive integer? What is its height?

4. (7 points) Give an example of a series of INSERT and EXTRACT-MIN operations on a Fibonacci Heap that will yield a heap of n keys with height $n - 1$.

5. (6 points)

Show the data structure that results and the answers returned by the FIND-SET operations in the following program. Use the linked-list representation with the weighted-union heuristic.	**for** $i \leftarrow 1$ to 16 **do**
	    MAKE-SET(x_i)
	end for
	for $i \leftarrow 1$ to 15 by 2 **do**
	    UNION(x_i,x_{i+1})
	end for
	for $i \leftarrow 1$ to 13 by 4 **do**
	    UNION(x_i,x_{i+2})
	end for
	UNION(x_1,x_5)
	UNION(x_{11},x_{13})
	UNION(x_1,x_{10})
	FIND-SET(x_2)
	FIND-SET(x_9)

6. (7 points) There is an image of "n by m" pixels. Originally all are white, but then a few black pixels are drawn. You want to determine the size of each white connected component in the final image. Pixels are judged as connected if they share a side.