ICS 311: Algorithms Spring 2025

Problem Set 1

Prof. Nodari Sitchinava *Due: Friday, January 17, 2025 at 4pm*

You may discuss the problems with your classmates, however **you must write up the solutions on your own** and **list the names** of every person with whom you discussed each problem.

Start **every** problem on a separate page, with the exception that Problems 2 can start on the same page as Problem 1 (Peer credit assignment).

1 Peer Credit Assignment (1 point extra credit for replying)

Please list the names of the other members of your peer group for this week and the number of extra credit points you think they deserve for their participation in group work.

- You have a total of 60 points to allocate across all of your peers.
- You can distribute the points equally, give them all to one person, or do something in between.
- You need not allocate all the points available to you.
- ***You cannot allocate any points to yourself!*** Points allocated to yourself will not be recorded.

2 Linked List via Arrays (30 pts)

You are hired by a company to implement a singly-linked list that will hold integers, using Java programming language. However, the boss of the company insists that you don't use any objects or structures in your implementation. You may use only integer arrays and individual integer variables.

Your implementaion should support the following operations:

- Insertion of the first integer element as the head of the list that will contain at most n elements throughout its lifetime.
- Keeping track of the 'current' element in the list (initially should be the head of the list)
- Advance 'current' to the next element in the list
- Reset 'current' to the head of the list
- Return the integer value of the 'current' element
- Insert a new value **following** the 'current' element in the list (should have no effect if n elements already had been inserted into the list)
- Delete the element **following** the 'current' element in the list (should have no effect if the 'current' element is the last one in the list)

The boss also insists that your implementation is efficient and **every operation should run in $O(1)$ time**, no matter what sequence of operations is performed. This is very important to the success of the company and they will not accept any code that is less efficient!

Fill in the missing code in the following definitions and explain why every one of your operations runs in $O(1)$ time:

```
public class List {

  private int ...

  private int[] ...

  /**************************************************************************
   * Initializes a linked list with integer 'val' as the head of the list
   * and identifies it as the current element
   * Ensures that no more than 'n' elements will be inserted into this list
   *************************************************************************/
  public List(int n, int val) {
    ...
  }

  /**************************************************************************
   * Advances the current element to the next element in the linked list
   *************************************************************************/
  public void advance() {
    ...
  }

  /**************************************************************************
   * Resets the current element to be the head of the list
   *************************************************************************/
  public void reset() {
    ...
  }

  /**************************************************************************
   * Returns the integer value of the current element
   *************************************************************************/
  public int value() {
    ...
  }

  /**************************************************************************
   * Deletes the element that follows the current element in the list
   * Has no effect if the current element is the last one in the list
   *************************************************************************/
  public void deleteNext() {
    ...
  }

  /**************************************************************************
   * Inserts 'val' into the list as the element that follows the current
   * one. Has no effect if n elements had already been inserted into the list
   * (not counting deletions)
   *************************************************************************/
  public void insertNext(int val) {
    ...
  }
}
```

3 Relative Growth Rates of Functions (30 pts)

Continuing in the style of in-class exercises, fill in the table for these pairs of functions with "Yes" or "No" in each empty box. Then for each row, justify your choice, preferably by showing mathematical relationships (e.g., transforming one expression into another, or into expressions that are more easily compared).

	$f(n)$	$g(n)$	O?	o?	Ω?	ω?	Θ?
e.	$2^{\log n}$	$\log^2 n$					
f.	$\sqrt{n}$	$n^{\cos n}$					
g.	$8n^2$	$4^{\log n}$					

4 Fun with Induction (40 pts)

Prove the following statements using **strong induction**:

(a) **(20 pts)** $\sum_{i=1}^{n} i \cdot i! = (n+1)! - 1$ for every positive integer n.

(b) **(20 pts)** $\sum_{i=1}^{n} i(i+1)(i+2) = \frac{n(n+1)(n+2)(n+3)}{4}$ for every positive integer n.

5 Fun with Logarithms (OPTIONAL - 0 pts)

(a) Prove the following identity:

$$n^{\frac{1}{\log n}} = 2$$

(b) Simplify the following expressions. **Show your work.**

- $n^{\log_3 9}$
- $n^{\log 8}$
- $n^{(1/\log_3 n)}$
- $n/(3^{\log_9 n})$

(c) Order the expressions in part (b) according to their asymptotic growth, from the lowest to the highest. **Justify your answers.**

6 Tiling with L-shaped Tiles (OPTIONAL - 0 pts)

A construction company is given a contract to tile bathrooms in a new apartment building in Kakaako. However, because of the customer's demands, the company is a bit stumped and hired you as a consultant to help them out.

The floor of **every** bathroom to be tiled is square-shaped, but bathrooms in different apartments are of **different** sizes. All you know is that each bathroom is of size $2^n \times 2^n$ inches for various **integers** n. For example, one bathroom might be of size 256×256 inches, another one might be 128×128 inches, and so on. The whole bathroom floor must be tiled, except for a square-shaped drain on the floor of size 1×1 inches. However, due to the unique design of the apartments, different bathrooms have the drain in **different** locations on the floor: some have them in a corner, others in the center, yet others a few inches away from the wall. All you know is that in each case, the drain will be some integer inches away from the walls. Moreover, the construction company must tile the floor using special tiles. Each tile is L-shaped, consisting of three 1×1-inch squares looking like this:

Given the strange shape of the tiles and so many possible locations of the drains, the first idea that the construction company had was cutting some of the tiles to fit the pattern. However, for esthetic reasons the customer wants as few tiles cut as possible. So the construction company hired you as a consultant to come up with an optimal solution which cuts the fewest tiles possible.

Prove that you can tile **every** bathroom in the building with these tiles without cutting **any** of them, regardless of where the drain is. For example, in the left figure below you can see an example of an 8×8-inch bathroom floor, with the drain pictured as a black square (the dotted lines are added to let you see that it is an 8×8 square). Next to it is an example of how it could be tiled with the L-shaped tiles without cutting any of them. *Hint: Use induction. If you need more space, you may use the back side of this page and the next page.*