

CS 314 FINAL REVIEW — ADD TO DIGITS LINKEDLIST

Linked Lists

Suppose you are given a `LinkedList` where each node represents a digit in a positive integer. The head of the list holds the most significant digit of the number while the tail of the list holds the least significant. This list of digits will be represented using the `DigitLinkedList` class.

Write an instance method called `addValue` for the `DigitLinkedList` class which will add an arbitrary, positive integer to the number represented by the list.

It is guaranteed that `this` is in a valid state when `addValue()` is called. This means that each `DigitNode`'s value will be between 0 and 9 inclusive. It also means that `first != null` (if the list stores the the value zero, then it would have a single `DigitNode` with a digit value of 0).

Examples of calls to `addValue()`. The result shown is the new state of `this`.

- `[3, 1, 1].addValue(3) => [3, 1, 4]`
- `[3, 1, 1].addValue(20) => [3, 3, 1]`
- `[4, 2, 9].addValue(10) => [4, 3, 9]`
- `[0].addValue(408) => [4, 0, 8]`
- `[1, 1].addValue(303) => [3, 1, 4]`

Use the following `DigitLinkedList` implementation.

```
public class DigitLinkedList {
    private DigitNode first;

    private static class DigitNode {
        // The nested DigitNode class.
        private int digit;
        private DigitNode next;
        private DigitNode(int d, DigitNode n){
            digit = d;
            next = n;
        }
    }
}
```

You may create new `DigitNode` objects using the provided constructor.

You may not create any other data structures.

Do not use any other Java classes or methods.

```
/* Pre: val >= 0; this list is in a valid state -> first != null
 * Post: this list will represent the sum of the previous state + val
 *       this list will still be in a valid state
 */
public void addValue(int val){
```