ECE 26400 Practica 2-b

1 Resources

Resources you **can** use:

- The C manual pages: in the terminal, type `man [function]` to open the manpages for the given function.
- Printed notes, as many as you want! Scratch paper and a pencil may also be useful.

You **cannot** use anything not mentioned above, including digital notes and online resources.

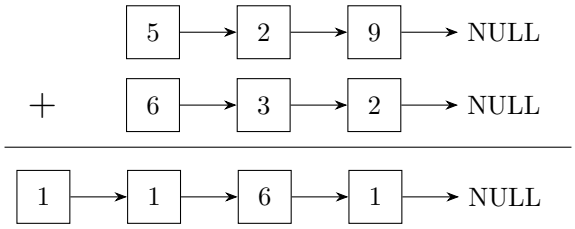
2 Problem: Big Int Addition

Your task is to implement a function that adds two non-negative "big integers". Big integers are integers of any size, represented by linked lists. Each node of the list stores a single base-10 digit, with the most significant digit first. Create the sum of the integers as a linked list, and return the head node of the list.

Write your solution in the `add` function in `practica_b.c`. You can add helper functions if you would like. The `add` function has two parameters `num1` and `num2` which are pointers (`ListNode *`) to the head of their respective big integer linked list. The `add` function should **create** a linked list that represents the sum and return a pointer to the head of the summed linked list.

The `ListNode` struct is defined in `practica_b.h`. Ensure you are familiar with its members before writing code.

Example 1: (More verbose explanation on next page)



`num1 = [5, 2, 9]` and `num2 = [6, 3, 2]` represent the integers 529 and 632. Adding these two integers results in 529 + 632 = 1,161. The returned sum list is `[1, 1, 6, 1]`.

Example 2:

`num1 = [1, 9]`, `num2 = [9, 8, 1]`. 19 + 981 = 1000. The returned sum list is `[1, 0, 0, 0]`.

2.1 Constraints

- The number of nodes in a list will range from `[1, 1000]` (both lists are always **non-null**).

2.2 Testing with Make

Run the below commands in your terminal to compile and test your code.

- `make test`x - Run test x where x is a number 1-10. Running `make testall` will run all tests.
- `make leak` - Run valgrind for memory leak checking.

3 Submission

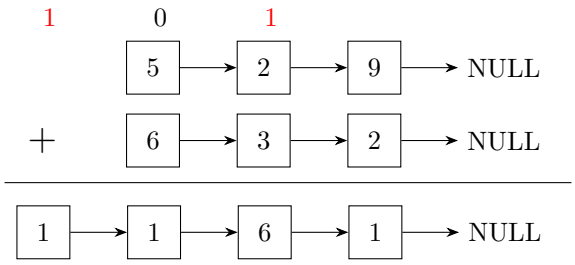
Submit only `practica_b.c` on Gradescope to run the autograder and get your score. Once you are finished, raise your hand and wait for a TA. They will check you off, after which you are allowed to leave.

4 Hints

- Traversing the list from tail to head to compute the resulting integer is difficult. Consider constructing a helper function that reverses a list, then traverse from head to tail on the reversed list.
- Consider storing a `carry` to help compute addition. See the examples below.

5 Examples

Example 1

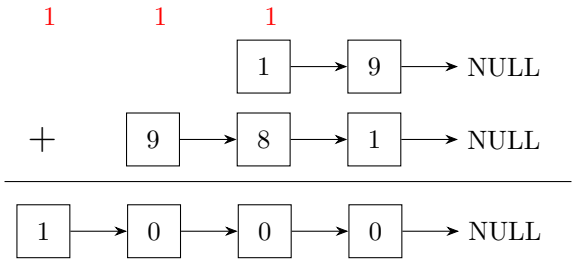


Input: `num1 = [5, 2, 9]`, `num2 = [6, 3, 2]`.

Output: `[1, 1, 6, 1]`

Explanation: Add the lists a digit at a time, starting from the least significant digit. $9 + 2 = 11$ with a `carry = 1`, resulting in the first node's value at 1. For the next digit, $2 + 3 +$ `(carry = 1)` $= 6$, and `carry = 0`. Continuing this process results in `[1, 1, 6, 1]`.

Example 2



Input: `num1 = [1, 9]`, `num2 = [9, 8, 1]`

Output: `[1, 0, 0, 0]`

Explanation: Starting at computing the least significant digit, compute $9 + 1 = 10$, resulting in `carry = 1` and a node is created with value 0. $1 + 8 +$ `(carry = 1)` $= 10$, `carry` is set to 1, and a node is added to the list with value 0. Continuing this process results in `[1, 0, 0, 0]`.