CSCI 320, Life Beyond Python, Spring 2022
Project 3: A doubly linked list

Project description

In this project you will implement a doubly linked list with bare-bones functionality. There are two parts, the public interface, which I specify, and the rest of the implementation, where you are free to express yourself.

The public interface

Here is the public interface for the list. Do not change the function signatures (i.e., the type of function, the types of its arguments, and the order of the arguments) since testing will assume this interface.

`LIST *new_list(const char *value)`

This function

- creates a new list,
- initializes its single node to the string `value`,
- updates the `head` and `tail` pointers in the list, and
- returns a pointer to the list.

`void prepend(LIST *list, const char *value)`

This function adds a new node at the head of the list. It

- creates a new node that contains the string `value`, and
- updates the `head` pointer in the list.

`void append(LIST *list, const char *value)`

This function adds a new node at the tail of the list. It

- creates a new node that contains the string `value`, and
- updates the `tail` pointer in the list.

`void delete_list(LIST *list)`

This function deletes a list and frees all its allocated memory.

What to do

Create your C functions in a file named `list.c`. Submit your file `list.c` and a corresponding Linux object file `list.o` to the autograder on Gradescope for grading. You do not need to submit `list.h`.