Asst0: String Sorter Readme

Mitchell Resnick (mjr327 | 162008981) & Liam Davies (lmd213 | 163004907)

# Overview

For this assignment, we decided to focus on modularity, code reusability, commenting, and general good practices. We have a few functions, all of which perform basic and necessary operations. We take an input stream, separated into tokens, and put those tokens into nodes, which are then inserted into a linked list in a sorted order.

# Functions

## getNextToken (Mitchell)

This function takes the input stream from argv[1]. There is a malloced int that keeps track of the position of the input stream. A function is defined as a contiguous array of characters A-Z or a-z. The eliminator is defined as any character that is not A-Z or a-z. The isalpha function was used, to make the code as robust as possible.

## stringComp (Liam)

This function takes two char\*s and compares them together in order to determine which string is greater. This function is used many times during the linked list insertion steps.

## createNode (Mitchell)

This function takes a char\* given by getNextToken and creates an empty node that can be put into a linked list. It sets the data portion of the node to the char\*, and the next pointer to be null.

## placeNode (Liam)

This function takes a node created with createNode and inserts it into the linked list using insertion sort.

## printLL (Mitchell)

This function simply steps through the list and prints out each node.

## freeLL (Liam)

This function frees the list by first freeing the char\*, and then freeing the node.