Bryan Erazo and Gal Zandani
CS 214
Assignment 0 - C Tokenizer

Readme

Description:
The program reads the input in the command line and makes sure that the number of inputs is correct. Then, the program allocates memory for the entire input which is a string using malloc and stores it in a local variable. It traverses the string and splits up the tokens in it according to the steps below. When it gets to the end of the string, the memory is freed.

The steps to tokenize and split up the string, going through each character and checking
If it is a letter, keep checking for letter/number. Then, it is a word.
    If this word is a C keyword, it is returned as the operator type of this keyword.
If it is a digit, it checks the following
    If it is a '0', hexadecimal or octal or decimal.
        If it is 'x' or 'X', it is hexadecimal and keep checking for 0-9|a-f|A-F.
        If it is 0-7, it is octal, and keeps checking for 0-7.
        If it is 8/9, it is decimal.
    If it is a 1-9, it stays a 0-9 digit, it is a decimal or if it is a '.' it's a float, and keep checking for digits.
    If it is a '.' or 'e(+|-)', it is a float, and keep checking for digits.
If it is a punctuation, we send it to the function that checks what kind of punctuation it is depending on its length. Starting with the longest option possible, of length 3, and going down it nothing is found to length 2 and then if needed to length 1.
    If it is a comment (// or /*...*/), it would skip that comment.
        If it is //, we skip until the end of the string.
        If it is /*, it would find it paired */ and go to the string after it.
If it is a space, we skip and start defining a new type of token.