# Programming Assignment: Compression v0

# Cleaning Up a Text File – Data Munging

This program focuses on file input/output.

### Introduction:

Analyzing text is a classic example of how a simple programming algorithm can quickly become messy. What do we do with capitalized words? Not too terrible to figure out. But what about punctuation? Normal text parsing uses a space delimiter, leaving behind all punctuation marks. If we are counting words, for example, and building a histrogram, many words will be appended with periods, commas, and quotations. Thus creating multiple versions of essentially the same word.

Before we write compression algorithms, we are going to pre-process our text files and create a clean version of all our input text files. This will make our compression algorithm comparisons more consistent, even though we are slightly altering our inputs.

The steps involved in Word Compression Version 0, for converting a given text ***input*** file into a ***clean*** text file are the following:

1. Examine the input file's contents using a space delimiter.
2. For each token, count the number of characters in the original token.
3. Then remove the following characters at the beginning or end of the token, as the case may be:

periods, commas, question marks, quote marks, open and close parentheses, and the String apostrophes (i.e.,‘s).

1. Write your filtered token to an output file. Be sure to append a space at the end of each token to rebuild the text file properly.
2. Sum the length of each modified token, not including spaces.
3. You do not need to worry about hidden characters in your file, such as <linefeed> or <end of file>. Your process will treat them as tokens, and rewrite them consistently to the output file.
4. Save your output file using the ***input*** file name, inserting “.clean” between the filename and the “.txt” extension.
5. Print both files names and both character counts to the console.

### Example File:

For example, here is the **input** file named example.txt with the following contents:

File Compression? It’s an algorithm created for compressing text data.

The **clean** text file named example.clean.txt should be:

File Compression Its an algorithm created for compressing text data

In the original file, this text contains 61 characters, not counting spaces. The “clean” file removes the question mark, apostrophe, and period, so it contains 58 characters.

mobydick.txt. starts with 976,240 characters, not including spaces. The clean version has 947,301 characters.

Note that the above statistics will not match up exactly with the file sizes reported in **Finder.** What are 2 reasons for this discrepancy?