=================================

= Assignment - Lite Lifting =

=================================

Description

===========

In lecture or reading, we have we have covered pipelines, regular expressions,

grep and some other text-processing utilities. These are the tried and

true tools of any \*nix'rs day to day life. From system administration to

website development and admin, even hardcore programmers know how to bend any

problem with the power of these utilities. As a mentor once chided me:

"If you're not using pipelines, you're working too hard."

So let's get to work learning how to work :)

Requirements, Assumptions, and Definitions

==========================================

Write a script, called LiteLifting.sh, it will read the first command line

argument ($1) as a data file name.

LiteLifting.sh will process the file in several different ways based on its

contents.

Your script may not use temporary files to store intermediate results. Instead,

make correct use of pipelines.

Each result should be produced by a single pipeline invocation (see the hint

below for how to store these results into a variable).

You may assume the following about the contents of the file:

1. The space characters will occur singly in the file. There will not

be any portion of the file with two or more spaces in sequence.

2. The file will contain only alphanumeric characters and punctuation,

white space (ASCII code 0x20, 32d), and newline characters.

Definition: a word in the file is any sequence of non-white space characters

bounded on either end by the beginning of the line, the end of the line, or

white space. For instance, consider a file containing:

Jack had a ball.

Jill.

The 4th word of the first line is ball. (including the period!); the second

word of the first line is had; and the second word of the second line is

nothing --- it doesn't exist.

Expected Output

===============

Your script should produce output of the following form:

This file has XXX characters in it.

This file has XXX characters not counting the newline.

This file has XXX empty lines.

This file contains XXX alphanumeric characters.

In all cases, the XXX should be replaced with the appropriate result. If a file does

not contain the specified data then the result should be reported as numerical 0

or an empty string depending on the context.

Hints

=====

This list of hints will hopefully help you understand how to get the values for

XXX in each of the lines of output above. If you're still confused, you can send

me an email!

1. wc anyone?

2. tr -d before wc ...

3. Expensive carrot? (An empty line is one that has no characters between the beginning of the line and the end of the line)

4. Remove newlines with tr, and characters not in 0-9a-zA-Z with sed :)

Slashes in the output are important (so we can see an empty string versus a whitespace).

5. Try replacing all newlines in the file with empty spaces before using grep, to capture

the phrase split across two lines.

You can use the exit status of the grep command directly (it exits with 0 if a match is found, 1 otherwise). Also, the -q option can be used to suppress the grep output.

Just to be clear, for the first part of processing the file, you will have a

separate pipeline for each of these questions. You are not expected to

construct one pipeline for all these answers.

You can set the value of a variable to the output of a pipeline with:

myCount=$(ls -1 |wc -c)

or alternatively (these are back-ticks),

myCount=`ls -1 |wc -c`

The first thing you should do is type up a file to test with! Then tackle these

one at a time --- we've tried to list them in order of complexity, and some

might build on the others.

Submission

==========

Submit your LiteLifting.sh script by uploading the file to Canvas.

Rubric [43 pts total]

=====================

[2 pts] Name your script LiteLifting.sh

[20 pts, 5 pts each] Correctly determines all output and matches output format above.

[5 pts] Capital Idea logic

[15 pts] No temporary files used, single pipelines for each result. Only 5

assignments used (no more than 5 '=' signs in your script). No partial credit.

[1 pt] Correct submission