File Operations in MIPS

I. Problem Statement

Write a MIPS Assembly Language program to produce the following statistics from an input file. A sample is attached: do not hardcode the file name in your source code.
How many: upper case letters, lower case letters, number symbols, other symbols, lines of text, and signed numbers?
Upload your source code, report, and any other required files here.

II. Approach

As part of the Course 5330 Computer Architecture, the Professor Dr. Richard Goodrum decided to test the skills of the students taking the course. In this Assignment basic file operations needed to be coded in Mips assembly language. An open source simulator/IDE for MIPS assembly language called MARS (version 4.5) was used to develop the assignment. Mars needed a Java 8+ JRE to be installed. I have used Java 11.0.12.

The problem statement required the filename to not be hardcoded. This was achieved by asking the user to input the filename from the user. However there was no specification regarding the location of the file or the size of the filename. These were kept by the developer for keeping the program simple. The file name provided cannot have a length greater than 32 characters. Also the file needs to be placed at the location of Mars application for it to be accessible from Mars.

The problem statement required "other symbols" and "signed numbers" but did not divulge what constituted the said items. This was left to the developers volition. Other symbols include all characters other than uppercase, lowercase and number symbols. Signed numbers include all numbers preceded by a positive or negative sign. Examples of signed numbers are +12. -0, +0, +67. Examples of non signed numbers are “abd”, “64” (not that there is no sign), "++d" (note that there is no digits).

When it came to lines of text, it was tricky as an empty document should give no of lines as 0 but a single line in the document with no “\n” character should yield no of lines as 1. Thus the formulae for no of lines should not be completely dependent on character “\n”. There must be a logic that takes into account if the document is empty or not as well.

III. Limitations

The file name is limited to a length of 32 characters. The file size is limited to 30 kilo bytes or 30720 characters. The interpretations of signed numbers and other symbols is left to the developer to decide and cannot be changed. The file used in this program must be placed at the location of the Mars application.

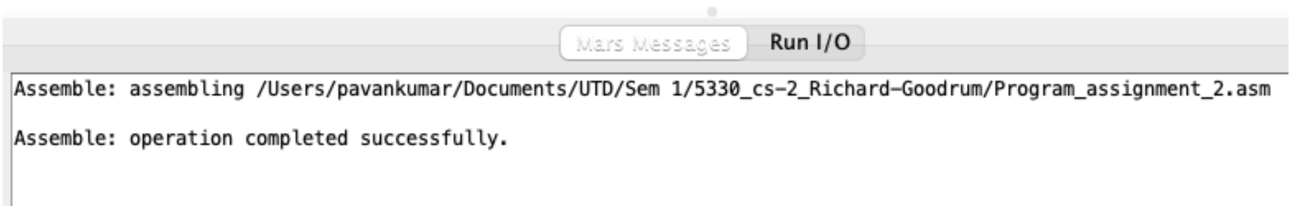
[Note: If facing difficulties opening the file, you may use sudo to run the Mars application as root user in linux or Mac systems. In windows, the Mars may be opened with Administrative privileges.]

IV. Solution

The program starts with a Descriptive prompt for the user’s edification on how to interact with the program. We obtain a file name from the user and perform analysis on the contents of the file. The data requested for such as upper case letters, lower case letters, number symbols, other symbols, lines of text, and signed numbers are computed and then displayed back as output on the user’s console.

V. Program Screenshots

Figure 1: Assembler output of the program is provided in this image. As seen it has successfully compiled else we can see errors at this position. At this point can run the program. In the next image, we'll discuss the output of the program.

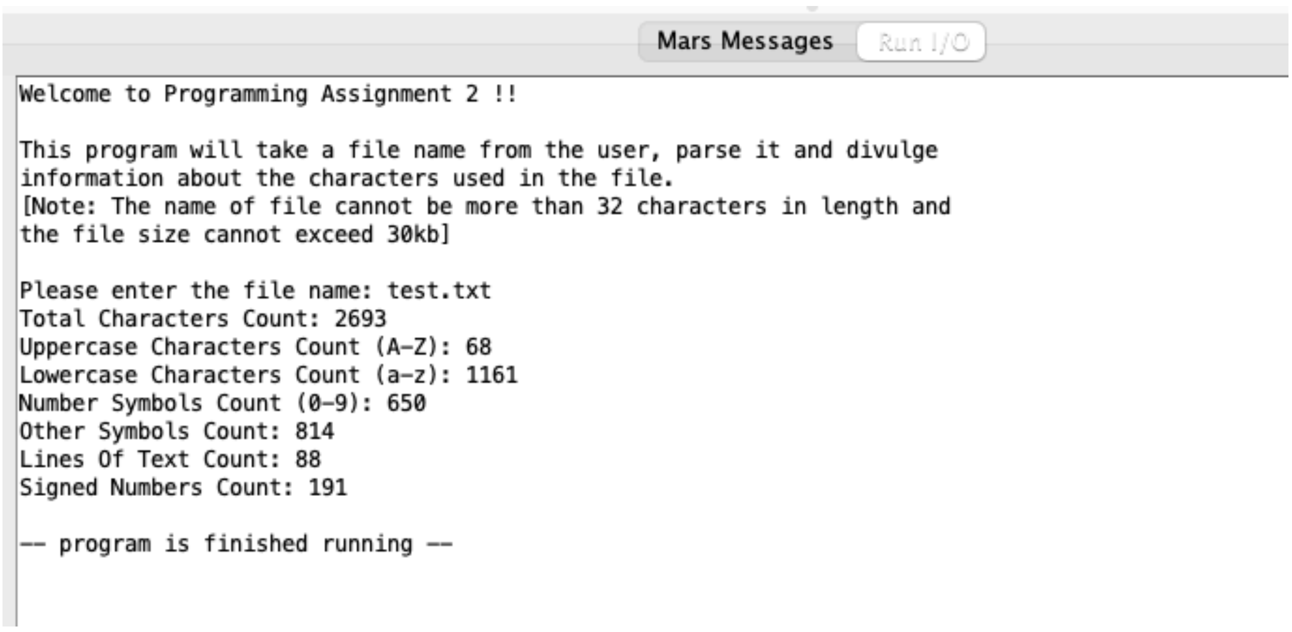


```
Assemble: assembling /Users/pavankumar/Documents/UTD/Sem 1/5330_cs-2_Richard-Goodrum/Program_assignment_2.asm

Assemble: operation completed successfully.
```

Figure 1: Mars Assembler Output

Figure 2: This is the console output as obtained from running the program in the MARS IDE. It shows the initial descriptive prompt to the user so that proper inputs can be provided. In this trial run, the sample file provided by the professor is used. The program then runs computing the counters that are outputted on the console as shown in the image.



```
Welcome to Programming Assignment 2 !!

This program will take a file name from the user, parse it and divulge
information about the characters used in the file.
[Note: The name of file cannot be more than 32 characters in length and
the file size cannot exceed 30kb]

Please enter the file name: test.txt
Total Characters Count: 2693
Uppercase Characters Count (A-Z): 68
Lowercase Characters Count (a-z): 1161
Number Symbols Count (0-9): 650
Other Symbols Count: 814
Lines Of Text Count: 88
Signed Numbers Count: 191

-- program is finished running --
```

Figure 2: Program Trial run and Output