CMPS 351: Spring 2014 Program #4
Date Assigned: Tuesday, April 1, 2014 Due: Tuesday, April 15, 2014

For this program, you will write a function that will compare two characters and determine which character comes first in alphabetical order. Because of the complexity of the rules given below, your function may not be able to determine whether a character is strictly "less than" another. Thus, your function will have more than two possible result values. For a function `compare(char a, char b);` your function should return 0 if a precedes b, 1 if a has the same precedence as b, 2 if a succeeds b, 3 if a alone is invalid, 4 if b alone is invalid 5 if both parameters are invalid.

The basic order of characters for English is: spaces, symbols, numerals and letters. Most punctuation (,.;:()[]{}<>'"!?) is ignored, however several symbols are still sorted (@#$%^&*+`~), also some symbols (- —/\) are considered spaces. All symbols have equal precedence, thus neither a "$" or a "%" can be said to precede the other in alphabetical order. Numerals and letters are sorted in the normal order (0123456789 and abcdefghijklmnopqrstuvwxyz.) However, the case of a letter is not significant (i.e. "A" and "a" have the same precedence and both come before "b" or "B".) Finally, if one or more of the parameters does not have a defined position in the order of precedence for English, then no comparison between the two characters can be made.

Your program will not need to interact with the console. Instead, you will use a test suite that has been provided for you. The test suite will call your function (compare) with the parameters ($a0 contains the value of the first character and $a1 contains the value of the second constant) and will wait for your function to return its results in $v0. The test suite will also tell you whether your function has correctly calculated the results. Finally, the test suite will also test to make sure that you are following assembly language conventions.

You should attach your code to the test suite for testing by running either:
Windows: copy /Y <Program #4 Function Name>.asm + "Program #4 - Test Suite.asm" <output>.asm
Unix: cat <Program #4 Function Name>.asm "Program #4 - Test Suite.asm" > <output>.asm
The result of either of these commands is a file that contains both sets of code, so that SPIM can load all of it at once. Your final submission should only include your function and not the test suite.

Your program should include appropriate comments indicating what the code should be doing and what registers are being used for. Please include your name and CLID in the program headers and include your CLID in the file names. Your programs should be turned in through Moodle before class starts on the due date. You should test your programs using the QT-SPIM simulator to ensure their functionality before submitting them.

Expected output:

```
Test #1 passed.
Test #2 passed.
Test #3 passed.
Test #4 passed.
Test #5 passed.
Test #6 passed.
Test #7 passed.
Test #8 passed.
Test #9 passed.
Test #10 passed.
Test #11 passed.
Test #12 passed.
Test #13 passed.
Test #14 passed.
Test #15 passed.
```

Objectives:

1. To introduce and practice building functions in the MIPS assembly language.