CMPS 351: Spring 2014 Program #5
Date assigned: Tuesday, April 15, 2014 Due: Thursday, May 1, 2014

For this project, you will write a MIPS assembly language program to perform a recursive insertion sort on a list of strings. This program extends program #4 as you will want to make use of the comparison function that you wrote to determine which of two strings precedes the other in alphabetical order.

A list is an object that contains two elements, first a pointer to some data and second a pointer to another list. Thus, an individual list object is 8 bytes in size. Below is some example code to sort a list of integers. You will need to extend the algorithm to support the alphabetical ordering (use your compare function) as well as to parse strings (1-byte character arrays.)

```
list* sort ( list* l ) {
      if ( l.next == null )      //  If this is the last element,
            return l;            //  then the list is already sorted.
      list* m = sort( l.next ); //  Otherwise, sort the rest of the list.
      l.next = null;
      m = insert( l, m );        //  Insert l.data into the sorted list.
      return m;
}
list* insert ( list* l, list* m ) {
      if ( m == null )           //  If this is the end of the list,
            return l;            //  then l is the only element.
      else if ( l.data < m.data ) {
            l.next = m;          //  If l is before m, make l the first
            return l;            //  element in the list.
      } else {
            m.next = insert( l, m.next );
            return m;            //  Otherwise, check the next element.
      }
}
```

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 (sort) with the parameter ($a0 contains the address of the first element in the list) 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 #5 Function Name>.asm + "Program #5 - Test Suite.asm" <output>.asm
Unix: cat <Program #5 Function Name>.asm "Program #5 - 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. After displaying the results, your program should exit cleanly. 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 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 review and practice with functions.
2. To introduce and practice working with the stack.
3. To introduce and practice working with arrays.
4. To introduce and practice working with pointers.
5. To introduce and practice working with objects.