CMPS 351: Spring 2013 Program #7
Date assigned: Thursday, March 7, 2013 Due: Thursday, March 21, 2013

This assignment asks you to write a MIPS function to use a binary search to find integers in an array. Your function will take two parameters, a tree to search through ($a0) and an integer to search for ($a1). If the queried integer is in the tree, your function should return the index of that integer. Most programming languages use 0 as the index of the first element in the list; however, for this program, you should use 1 for the first element in the list. If the queried integer is not in the tree, then you should return 0. If there is a problem with the input, your function should return -1.

The tree is provided as an object with two parts. The first element in the tree is an integer that contains the size of the tree. The rest of the tree is composed of a sorted list of integers, such that smaller numbers appear before larger numbers.

The primary goal of this program is to practice moving through memory in a non-linear fashion. Thus, using a linear search algorithm will not be acceptable. There are several ways to impose a tree structure on top of a sorted array and you may choose any one of them to implement.

This program has a test suite similar in complexity to that of program #6. However, there will be differences between the test suite provided for initial testing and the one used for grading. Thus, you should make sure that your program handles all the tests in the suite and you will also want to write additional tests to ensure that your program correctly handles all possible cases.

Your program should include appropriate comments indicating what the code should be doing and which registers are being used. 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 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.
```

Objectives:

1. To practice with memory access in the MIPS assembly language
2. To practice non-linear address calculations for the MIPS architecture.
3. To practice defensive programming.

Point Values:
Total. 150pts