CENG311 **Fall 2019**

PROGRAMMING ASSIGNMENT 1

Due date: 21.11.2019 10:00 am

In this assignment, you are required to implement some binary search tree operations in MIPS assembly language. You will use SPIM simulator [1] to develop and test your code.

First of all, you will implement a binary search tree with nodes which have the following structure:

Byte Address:	Contents
X:	Value of the node
X + 4:	Address of left child
X + 8:	Address of right child
X + 12:	Address of parent

Each node of your tree should be stored in 16-byte address which hold the value of the node, the address of the left child, the address of the right child, the address of the parent node respectively. When there is no left child or right child or parent, the memory address is 0.

The property that makes a binary tree into a binary search tree is that for every node, X, in the tree, the values of all the keys in its left subtree are smaller than the key value in X, and the values of all the keys in its right subtree are larger than the key value in X (More information on [2]).

Procedures to be Implemented:

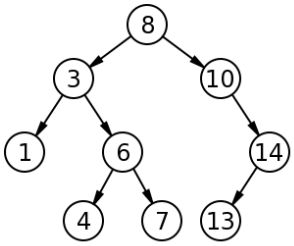
build (*list, tree*)
This procedure will construct a binary search tree data structure from a list of integers. The address of the first integer is in the ***list*** argument (assume the list is terminated by a special value, -MAXINT (the negative integer with the largest possible absolute value)), and the address of the place where the procedure should create the data structure in **tree** argument (assume there is enough space for the tree structure).

insert (*value, tree*)
This procedure will create and put a new node (the value is given in $a0 register) to the binary search tree (the address of the root node of the tree is in ***tree*** argument). The procedure will require new space in memory for the new tree node, which can be obtained with the MIPS system call *sbrk.* The address of the location where the new node was inserted should be stored in $v0 register.

find (*value, tree*)
This procedure will try to find the value (given in $a0 register) in the binary search tree (the address of the root node of the tree is in ***tree*** argument). The search result should be stored in $v0 register (If found $v0 = 0, if not $v1 = 1). If the value is found in the tree, $v1 should should contain its address.

findMinMax (*tree, function*)
This procedure will find the minimum or maximum value in the binary search tree (the address of the root node of the tree is in ***tree*** argument) according to the value of the ***function*** argument. If ***function*** equals to 0, the procedure will find the minimum; if ***function*** equals to 1, the procedure will find the maximum. The value of the node should be stored in $v0 register, the address of the node should be stored in $v1 register.

print (*tree*)
This procedure will print the binary search tree (the address of the root node of the tree is in ***tree*** argument) to the screen. Each line in the screen output will have one level of the tree. If there is no children (left or right), there will be “x” character to represent no node.
Example binary search tree:



Example output:
8
3-10
1-6 x-14
x-x 4-7 x-x 13-x

Assumptions:

- The arguments to the procedures are stored in $a registers, the first is in $a0, the second is in $a1, and so on.
- Only valid arguments are passed into the procedures. You do net need to check the arguments for their validity.
- The input and output lists have maximum 100 characters-length.

Requirements:

- The values in all $a registers should be as they were at the time of call at the end of the procedure.
- Your program should have a main procedure which provides a menu to demonstrate your work. The menu should include the list of the procedures and one can select to execute a procedure by providing necessary inputs. The result (return value(s)) should also be written to the console at the return of the procedure.
- Your code should be ready for a test program (provided by your TA) which checks flow and result of the procedures.
- You are required to submit the source code and a report that includes implementation details and screenshots of your sample executions.
- You need to work individually, no group work is allowed.
- No late homework will be accepted.

Submission: You are required to submit your commented source code and report to CMS. Please create a compressed file including all source files and report; and name it as yourstudentnumber_P1.zip (e.g. If your student number is 201812345678, the file name must be 201812345678_P1.zip).

[1] http://spimsimulator.sourceforge.net/
[2] Data Structures and Algorithm Analysis in C, Mark A. Weiss, Addison Wesley