CSE1729 - Introduction to Programming October 26, 2020

Laboratory Assignment 7

Objectives

- Work with trees
- Use Binary Search Trees

Activities

Binary Trees

Recall the following code from the lecture slides:

```
(define (make-tree value left right)
  (list value left right))

(define (value T) (car T))
(define (right T) (caddr T))
(define (left T)  (cadr T))
```

1. Write a SCHEME function, named `(tree-size T)`, which takes a tree node, `T`, and returns the size (i.e. the number of nodes) of the tree rooted at `T`.
2. Write a SCHEME function named `(tree-depth T)` which takes a tree node, `T`, as a parameter and returns the depth of the tree rooted at `T`. For the purpose of this exercise, the depth of a tree rooted at `T` is one plus the maximum of the depths of its two children.
3. Define a SCHEME function named `(count-pred P tree)` which given a binary tree and predicate function, `P`, applies the predicate to each of the values in the tree and returns the number of values for which the predicate returns `#t` (true).
4. Define a SCHEME function named `count-one-child` which returns the number of internal nodes of a binary tree which have exactly one child.

Binary Search Trees

```
(define (insert x T)
  (cond ((null? T) (make-tree x '() '()))
        ((eq? x (value T)) T)
        ((< x (value T)) (make-tree (value T)
                                    (insert x (left T))
                                    (right T)))
        ((> x (value T)) (make-tree (value T)
                                    (left T)
                                    (insert x (right T))))))
```

5. Define a SCHEME function named `(invert-bst T)` which, given a Binary Search Tree (BST) *inverts* that BST. That is, the function returns a binary tree in which has the property that all values at nodes in the left subtree are greater than the value at the root node and all values at nodes in the right subtree are less than the value at the root node. Hint: There is an extremely simple way to write this function.

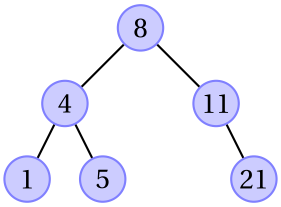

Figure 1: A binary search tree with 6 nodes, 4 of which are 5 or larger.

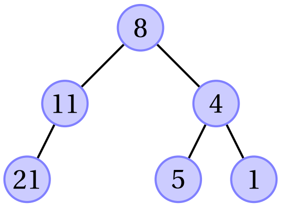

Figure 2: An inverted binary search tree with 6 nodes.