CS 304: Programming Languages
Assignment 6

November 14th, 2012

Due: November 28th, 2012 before the beginning of the class period

1. a) (1.5 pts) Write scheme code to implement a binary search tree and the functions to operate on the tree. The tree is manipulated by the following functions :

   `(insert key tree)` to insert a new node with the specified key in the tree and returns the resulting tree

   `(remove key tree)` to delete a node from the tree, if that node exists in the tree. If the node does not exist in the tree, no removal should take place

   `(find key tree)` returns #t or #f depending on whether key is in the tree

   `(isEmpty? tree)` returns #t or #f depending on whether the tree is empty or not

   `(height tree)` returns the height of the tree

   `(makeListFromTree tree)` returns a list of the nodes of the tree in sorted order

   b) (0.5 pts) Write a comprehensive 'driver program' that will test and display the workings of all of your functions from part a).

2. (1 pts) Define a Scheme procedure `flatten` that takes a complex list and converts it into a simple list. For example,

   `(flatten '(1 2 (3 (4 5) 6 (7 (8) 9))))` yields `(1 2 3 4 5 6 7 8 9)`

3. (1 pts) Define a Scheme procedure `quicksort` that takes a simple list of numbers and returns a list with the numbers in ascending order using the quicksort algorithm.

4. (1 pts) Define a Scheme procedure mirror-image that takes a list (either simple or non-simple) and reverses all of the elements in the list, including those in any sub-lists that the input list may have. For example

   (mirror-image '(c o c o n u t)) yields (t u n o c o c)

   (mirror-image '(a b (c d (e f) n) u t)) yields (t u (n (f e) d c) b a)

5. (1 pts) Define a Scheme predicate palindromic that tests whether a given list is palindromic – your function must work also on complex (nested) lists by checking each internal list for palindromes as well. For example:

   (palindromic '(a (b c) d (b c) a) yields #T

   (palindromic '(a (b c) d (b b) a) yields #F

6. (2 pts.) Write a Scheme function `powerset` that takes a simple list as input and returns the powerset of the list. Assume the input list as a set, and generate the powerset. A powerset is defined as a set of all subsets of a set. Assume that all of the elements of the input list are distinct. HINT: powerset of {a, b, c, d,...} is equal to union of powerset of {b, c, d,...} and 'a' appended to each element of powerset of {b, c, d,...}. The ordering of the sets in the powerset is not important. For example,

   `(powerset '(1 2 3))` → yields `(() (1) (2) (3) (1 2) (1 3) (2 3) (1 2 3))`

7. (2 pts.) Write a Scheme function `permutations` that takes a list and produces a list with list elements being all permutations of the given list. Assume the input list as a set and that it contains no duplicates. For example,

   `(permutations '(1 2 3))` yields

   `((1 2 3) (1 3 2) (2 1 3) (2 3 1) (3 1 2) (3 2 1))`

Turn In Procedures:

You may turn in your assignment via email (bkerkez@yahoo.com) or in person: on paper or on any computer media. If you wish to turn in your assignment via email, your file name should be in the format

***lastName_FirstName*_AssignmentX.xxx**

For example, John Smith's assignment 6 turn in file would be named "Smith_John_Assignment6.xxx".

If you chose to turn in your assignment via email, please make sure that the subject of the email message reads "**CS304_Assignment**". Please print a paper copy of your assignment and turn it in as well.

Please make sure that I receive your assignments before the due date and time. Late assignments will be accepted for 24 hours after they are due and will be penalized by a 50% deduction. No assignments will be accepted after the 24 hour period expires.