Bilkent University
Computer Engineering Department
CS 315 - Programming Languages 2017-2018 Spring

Homework III - Programming in Scheme

Assigned: May 7, 2018
Due: May 14, 2018, 23:59

1. *Setify*

   Write a Scheme function, called `setify`, that takes a list of atoms, and returns a new list that contains only the unique atoms in the list. You can define extra functions if you need.

   Some example calls are given below:

```
> (setify '())
()
> (setify '(a))
(a)
> (setify '(2 a 1.6))
(2 a 1.6)
> (setify '(a b c a b c d a d))
(b c a d)
> (setify '(2 a 1.6 2 1.60))
(a 2 1.6)
```

2. *join*

   Write a Scheme function called `join`, that takes two sorted lists and returns the sorted list of the elemets of the arguments, without repetition. Try to make it as efficient as possible.

   Sample runs:

```
> (join '(1 3 5) '())
(1 3 5)
> (join '() '(5 6 7))
(5 6 7)
> (join '(3 5 9) '(2 4 5 6 8 10))
(2 3 4 5 6 8 9 10)
```

3. *BST insert*

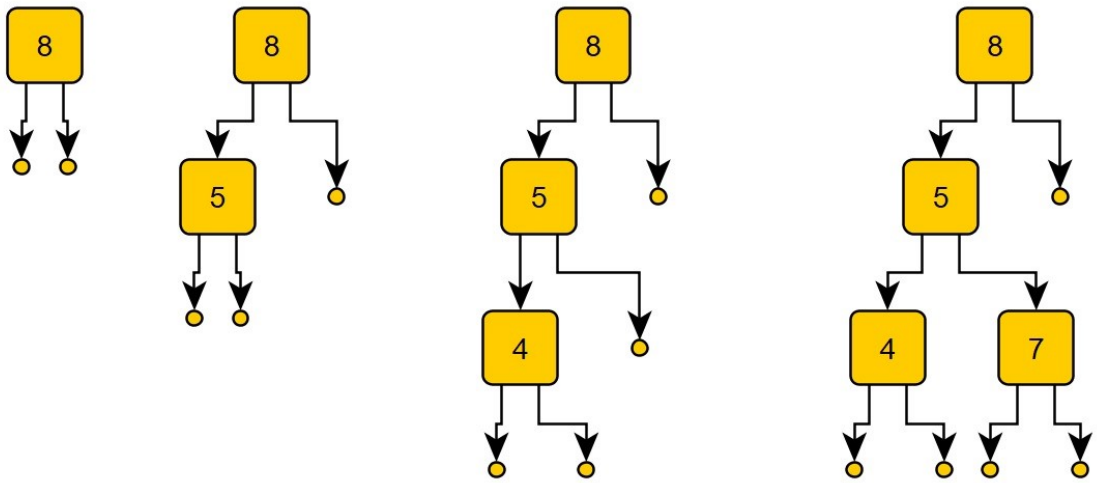
   A binary search tree is a rooted search tree, whose internal nodes store a *value*, and two subtrees, namely *left subtree* and *right subtree*. The key invariant of a binary search tree is that for every node, all the values in the left subtree are smaller than or equal to the *value*, and all of the values in the right subtree are larger than the *value*.

   That structure can be implemented in Scheme using a three-element list: first element stores the value, second stores the left tree and third stores the right tree.

   Write a Scheme function called `insert-bst`, that takes a number to be inserted, and a valid binary search tree. The result of the function should be the binary search tree with item inserted.

Sample runs:

```
> (insert-bst 8 '())
(8 () ())
> (insert-bst 5 '(8 () ()))
(8 (5 () ()) ())
> (insert-bst 4 '(8 (5 () ()) ()))
(8 (5 (4 () ()) ()) ())
> (insert-bst 4 (insert-bst 5 (insert-bst 8 '())))
(8 (5 (4 () ()) ()) ())
> (insert-bst 7 (insert-bst 4 (insert-bst 5 (insert-bst 8 '()))))
(8 (5 (4 () ()) (7 () ())) ())
```



Logistics

- You may define extra functions if needed.
- You may use the tutorials as a reference, but do not derive your programs or samples from these tutorials. If you do so, your grade will be negatively affected.
- Place your programs under different files each having the question number, with your last name and name. For example, `q1_lastname_name.scm`, `q2_lastname_name.scm`, `q3_lastname_name.scm`.
- Write a report that also contains your sample programs, and the results of interpreter output of sample executions.
- Put your report and source files into a folder named `lastname_name` and make a zip of the folder with name `lastname_name.zip`. Then, e-mail this file to the TA, Gizem aylak `(gizem.caylak -at-bilkent.edu.tr)`.
- Collaboration on the homework is not allowed.
- Your programs should be compliant with the `/usr/bin/scheme` installation on the dijkstra machine.

Suggestion: Do not postpone the execution of your programs to the last minute! The disjkstra machnine might be overloaded, then.