COMP 335: Principles of Programming Languages
Assignment 0: Racket Basics

Due date: Tuesday September 9, 2014, at 11:59pm

Collaboration policy

- You may discuss the programming problems with each other
- You may not share or look at each other's code
- You must give credit to any outside resources used (for example, non-course textbooks or wikipedia)
- Answers to written questions must be your own work

Handin instructions

Hand in the following files:

<first initial><lastname>_hw0_1.rkt – DrRacket definitions file for Programming problem 1.
<first initial><lastname>_hw0_1.rkt – DrRacket definitions file for Programming problem 2.
<first initial><lastname>_hw0.pdf – PDF file with your answers to the written problems,.
<first initial><lastname>_hw0_README.txt – (optional) Text file with anything else you wish to tell me.

Programming (40 points)

1. (20 points) Implement insertion sort algorithm for a list of numbers. Your code should adhere to the following template, filling in the missing `<procedure>`, and should pass all the given tests:

```
#lang plai-typed

(define (insertion-sort [x : (listof number)]) : (listof number)
     <procedure>
)

#|
   Hint: you may want to implement a recursive helper function
   which takes as input a sorted (listof number) and a new number
   to add and returns a (listof number) with the the new number
   in the correct place.
|#

; Tests:
(test (insertion-sort (list 1 2 3)) (list 1 2 3))
(test (insertion-sort (list 3 2 1)) (list 1 2 3))
(test (insertion-sort (list 1 3 1)) (list 1 1 3))
(test (insertion-sort (list 1)) (list 1))
(test (insertion-sort empty) empty)
```

2. (20 points) Here's another way to sort a list of numbers: build a binary search tree from the numbers to be sorted, and then traverse the tree in-order to find the sorted order.

For example, the following list:

```
(list 5 4 7 9 8 3)
```

can be used to build the following binary tree:

```
(node 5 (node 4 (node 3 (leaf) (leaf)) (leaf)) (node 7 (leaf) (node 9
(node 8 (leaf) (leaf)) (leaf))))
```

which has the data-type defined as follows:

```
(define-type Tree
  [node (n : number) (left : Tree) (right : Tree)]
  [leaf])
```

For this problem, you will write code to traverse a binary search tree (such as the example) and return (listof numbers) list in sorted order. You do not need to write code to build the search tree, but you will need to write your own test cases.

Your code should adhere to the following template, filling in the missing `<procedure>`:, and test cases. You may also define helper functions if necessary.

```
#lang plai-typed

(define-type Tree
  [node (n : number) (left : Tree) (right : Tree)]
  [leaf])

; Helper functions (if necessary)

(define (tree-to-list [t : Tree]) : (listof number)
  <procedure>)

; Test cases:
```

Writing (20 points, 5 points each)

1. Rewrite the following s-expression so that it uses only binary arithmetic operations. You do not need to draw a tree, just write the s-expression. Show each step of your work:

```
     (* 1 2 3 (+ 4 5 6))
```

2. Explain in your own words (2-3 sentences max) why the expression `(first (list 1 2))` evaluates to a number, but `(rest (list 1 2))` evaluates to a list.

3. Plai-typed has a procedure called `second` which returns the second element in a list.

   a. Write an equivalent procedure using `first` and `rest`. Explain your work.

   b. Consider the following s-expression:

   ```
   ((1 (2 3)) (4 5) 6)
   ```

   What is the second element of this s-expression list? Explain.
   (Note: Actually running this procedure requires a lot of type-casting. Just reason it out.)

4. In order to read and evaluate an arithmetic expression such as `3 * 4 + 2 * 3` in C++, Python, or Java, it is necessary to follow the rules for arithmetic operator precedence. Without knowing that multiplication operations should be performed before addition, it would be unclear how to evaluate this expression.

Now consider arithmetic expressions (addition and multiplication only) which are written in s-expression form. Are there arithmetic s-expressions which are unclear to evaluate without using order of operations rules? Why or why not? Use examples to support your argument.

Total score for Assignment 0: 60 points