**Lab 4**

**Recursion and Set Operation in Racket**

This is an individual assignment. In this lab assignment, you will work experimentally with the DrRacket language on Recursions and Set Operations.

In Racket sets can be represented as lists. However, unlike lists, the order of values in a set is not significant. Thus both (1 2 3) and (3 2 1) represent the same set.

\*\*\*For the following questions, **you may assume that lists are used as the input data to represented sets, but they may contain duplicates, and you need to address this issue**. However, you can assume that they contain only atomic values (numbers, string, symbols, etc.).

\*\*\* You must use recursion, and not iteration. You may not use side-effects (e.g. set!).

**Part I** (5 points per question)

1. Write a Racket function(subset? L1 L2) that tests whether L1 ⊆ L2. L1 is a subset of L2 if every element of L1 is also a member of L2. For example:

(subset? '(1 2 3) '(3 2 2 1))---> #t

(subset? '(1 2 2 3 3) '(1 3 2)) ---> #t

(subset? '(1 1 2 3) '(1 2 3 4 5 6)) ---> #t

(subset? '(1 2) '())---> #f

\*\*\*Use the function(member? x L)as a helper function in your implementation.

1. Write a Racket function (set-equal? L1 L2) that tests whether L1 and L2 are equal. Two sets are equal if they contain exactly the same members, ignoring ordering (or in other words, two sets are equal if they are a subset of each other). For example:

(set-equal? '(1 2 3) '(3 2 2 1)) ---> #t

(set-equal? '(1 3 3 2) '(3 2 1)) ---> #t

(set-equal? '(ryan susan john) '(susan john ryan)) ---> #t

1. Two common operations on sets are union and intersection. The union of two sets is the set of all elements that appear in either set (with no repetitions). The intersection of two sets is the set of elements that appear in both sets.

Write Racket functions (union S1 S2) and (intersect S1 S2) that implement set union and set intersection.

Test cases:

(union '(1 2 3) '(3 2 2 1)) ---> (1 2 3)

(union '(1 2 3 2) '(3 4 5)) ---> (1 2 3 4 5)

(union '(a b c c) '(1 3 2 1)) ---> (a b c 1 2 3)

(intersect '(1 2 3 3) '(2 3 2 1)) ---> (1 2 3)

(intersect '(1 2 1 3) '(1 4 1 5 1 6)) ---> (1)

(intersect '(1 2 1 3) '(2 3 4 5 2 6)) ---> (2 3)

The ordering of the elements in your answer may differ from the above.

**Submission**

Prepare a single Racket program file (lab4.rkt) containing definitions of all the requested functions. Please make sure to use the requested function names and use comments to explain the parts that are hard to understand. Submit the file on Canvas.