Assignment 05

Due: Wednesday, June 14th , 2017 at 10am

- You must provide the data definition and template in your solutions **only** when the question specifically indicates they are required for compound data types described in the question. If you create any additional data types that are beyond the question description, your program file should include a data definition and a template for each additional data type.
- If you include a template in your solution, the template should appear as comments.
- You may want to include defined constants to help reduce the writing for the examples and tests
- Unless otherwise indicated by the question you may only use the built-in functions and special forms introduced in the lecture slides from CS115 up to and including the modules covered by this assignment. A list of functions described in each module of the lecture slides can be found at https://www.student.cs.uwaterloo.ca/~cs115/built_in.
- Use the design recipe when writing functions (and helper functions) from scratch.
- Download the interface file from the course Web page to ensure that all function names are spelled correctly, and each function has the correct number and order of parameters.
- Read each question carefully for restrictions.
- Test data for all questions will always meet the stated assumptions for consumed values.
- Do not copy the purpose directly from the assignment description. The purpose should be written in your own words and include references to the parameter names of your functions.
- You may post general assignment questions using the discussion forum on Waterloo LEARN. Choose Connect -> Discussions. Read the guidelines for posting questions. Do NOT post any code as part of your questions.
- The solutions you submit must be entirely your own work. Do not look up either full or partial solutions on the Internet or in printed sources.
- Do not send any code files by email to your instructors or tutors. Course staff will not accept it as an assignment submission. Course staff will not debug code emailed to them.
- Check Markus and your basic test results to ensure that your files were properly submitted. In most cases, solutions that do not pass the basic tests will not receive any correctness marks.
- Any string or symbol values must **exactly** match the descriptions in the questions. Any discrepancies in your solutions may lead to a severe loss of correctness marks.
- Read the course Web page for more information on assignment policies and how to organize and submit your work. Follow the instructions in the Style Guide. Your solutions should be placed in files `a05qY.rkt`, where `Y` is a value from `1` to `3`.

Language level: Beginning Student
Coverage: Module 5

1. Using structural recursion, write a function called `remove-occurrence` that consumes a list of strings `los` and a string `find`, and produces a list of all strings in `los` that do not contain `find` (i.e., `find` is not a substring of any of the strings in the produced list.)

   For example:
   - `(remove-occurrence (cons "catalog" (cons "bobcat" (cons "bill" (cons "phone" empty)))) "cat") => (cons "bill" (cons "phone" empty))`

Assignment 05

Due: Wednesday, June 14th , 2017 at 10am

2. Using structural recursion, write a function called `factors`, that consumes a list of non-zero natural numbers, `lon`, and produces a list of all the factors of the sum of all the numbers in the list. The values in the list produced should appear in the same order as they appear in `lon`.

   For example:
   - `(factors (cons 4 (cons 3 (cons 1 (cons 2 empty))))) =>`
     `(cons 1 (cons 2 empty))`

3. A tautogram list is a list of non-empty strings where all the strings in the list start with the same letter. An empty list is a tautogram list.

   Write a Racket function `tautogram-list?` that consumes a list of strings, and produces `true` if the list is a tautogram list, and `false` otherwise.

   Note that the strings are NOT case sensitive, i.e. "abc" is equivalent to "ABC".

   For example:
   - `(tautogram-list? (cons "blue" (cons "skies" empty))) => false`
   - `(tautogram-list? (cons "apple" (cons "ale" (cons "a" empty)))) => true`
   - `(tautogram-list? (cons "Ale" (cons "ale" empty))) => true`