CS115 **Assignment 04** Fall 2019

**Due: Friday, October 11 @ 11:30 PM (No late submissions)**

Assignment Guidelines:

- Solutions to these questions are expected to follow the requirements of the Style Guide (https://www.student.cs.uwaterloo.ca/~cs115/coursenotes1/styleguide.pdf). This includes all relevant design recipe elements, proper use of constants, and proper use of helper functions.
- Submission details:
  - Solutions to these questions must be placed in files **`a04q1.rkt`**, **`a04q2.rkt`**, **`a04q3.rkt`**, respectively, and must be completed in Racket.
  - All solutions must be submitted through MarkUs. Solutions will **not** be accepted through email.
  - Verify your basic test results using MarkUs to ensure that your files were submitted properly and are readable on MarkUs. *Note, however, that passing the basic tests does not guarantee that you will pass all our correctness tests.*
- 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.
- Restrictions:
  - You may only use the built-in functions and special forms introduced in the lecture slides up to and including the module covered by this assignment. A list of these functions can be found on the Assignments web page: https://www.student.cs.uwaterloo.ca/~cs115/#allowed
  - Read each question carefully to see if any additional restrictions apply.
  - Test data for correctness tests will always meet the stated assumptions for consumed values.
- 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.

Plagiarism: The following applies to all assignments in CS115.

All work in CS 115 is to be done individually. The penalty for plagiarism on assignments (first offense) is a mark of 0 on the affected question and 5 marks off the final grade, consistent with School of Computer Science policy. In addition, a letter detailing the offense is sent to the Associate Dean of Undergraduate Studies, meaning that subsequent offenses will carry more severe penalties, up to suspension or expulsion.

To avoid inadvertently incurring this penalty, you should discuss assignment issues with other students only in a very broad and high-level fashion. Do not take notes during such discussions, and avoid looking at anyone else's code, on screen or on paper. If you find yourself stuck, contact the ISA or instructor for help, instead of getting the solution from someone else. Do not consult other books, library materials, Internet sources, or solutions (yours or other people's) from other courses or other terms.

Read more course policies at: https://www.student.cs.uwaterloo.ca/~cs115/#policies

**Language level:** Beginning Student
**Coverage**: Module 4

**Due: Friday, October 11 @ 11:30 PM (No late submissions)**

**Question 1:** Complete the Racket function **`append-with-separator`** that consumes a list of strings, **`los`**, and a (possibly empty) string of arbitrary length, **`sep`**, and produces a string of all elements of **`los`** with **`sep`** appended after each one.

For example:

- `(append-with-separator empty " ") => ""`
- ```
  (append-with-separator (cons "programming" (cons "is"
    (cons "fun?" empty))) "...")
  => "programming...is...fun?..."
  ```
- ```
  (append-with-separator (cons "Beep" (cons "Boop"
    (cons "" (cons "Bop" empty)))) ", ")
  => "Beep, Boop, , Bop, "
  ```
- ```
  (append-with-separator (cons "Hi" (cons "World" empty)) "")
  => "HiWorld"
  ```

**Question 2:** Complete the Racket function **`keep-multiples-of-three-or`** that consumes a list of integers, **`loi`**, and a non-zero natural number, **`mult`**, and produces a list of all elements in **`loi`** that divide evenly by 3 or **`mult`**, but not both. Note that 0 is a multiple of every number.

For example:

- `(keep-multiples-of-three-or empty 2) => empty`
- `(keep-multiples-of-three-or (cons 3 empty) 3) => empty`
- ```
  (keep-multiples-of-three-or
    (cons 9 (cons 6 (cons 11 (cons -4 (cons 0 empty))))) 6)
  => (cons 9 empty)
  ```
- ```
  (keep-multiples-of-three-or (cons 105 (cons 6
    (cons 25 empty))) 5) => (cons 6 (cons 25 empty))
  ```
  ```

**Due: Friday, October 11 @ 11:30 PM (No late submissions)**

**Question 3:** Complete the Racket function **`convert-to-types`** that consumes a list of strings, **`values`**, where each element is some Racket value that has been converted to a string (e.g. the string **`"-3.4"`** would correspond to the Racket numeric value **`-3.4`**). The function produces a list of the same length as **`values`** where each element of **`values`** is replaced by a string representation of the type of its original Racket value, using the following rules:

- Possible types are **`"Nat"`**, **`"Int"`**, **`"Num"`**, **`"Bool"`**, and **`"Str"`**.
- If the string can be converted to a number then use one of **`"Nat"`**, **`"Int"`**, or **`"Num"`**. You may find it useful to review the Racket documentation of the built-in function **`string->number`** before attempting the question.
  Recall that the set of natural numbers, N, is a subset of all integer numbers, Z, which in turn is a subset of the real numbers, R. Pick the most restrictive group that the value belongs to, e.g. replace **`"-5"`** with the type **`"Int"`** and **`"0"`** with the type **`"Nat"`**.
- If the string represents any of the Boolean values **`"true"`**, **`"false"`**, **`"#true"`**, **`"#false"`**, **`"#t"`**, or **`"#f"`**, then use **`"Bool"`**.
- Use **`"Str"`** otherwise.

For example:

- **`(convert-to-types empty) => empty`**
- ```
  (convert-to-types
    (cons "-3.4" (cons "#true" (cons "cs115" (cons "2"
      (cons "false" (cons "#t" empty)))))))
    =>
  (cons "Num" (cons "Bool" (cons "Str" (cons "Nat"
    (cons "Bool" (cons "Bool" empty))))))
  ```
  ```