CS 115 – Spring 2017
Assignment 04
Due: Wednesday, June 07 at 10:00 a.m.

- 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 in the question, you may use only 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
- For this and all subsequent assignments, you are expected to use the design recipe when writing functions from scratch, including helper functions.
- 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 in your function headers.
- 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 other course staff. Course staff will not accept it as an assignment submission. Course staff will not debug code emailed to them.
- 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.
- 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. Basic tests results will catch many, but not necessarily all types of errors.
- 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 `a04qY.rkt`, where `Y` is a value from `1` to `3`.

Language level: Beginning Student
Coverage: Module 4

Be very careful in reading the data definitions and the associated questions in order to fully understand their relationship.

CS 115 – Spring 2017
Assignment 04
Due: Wednesday, June 07 at 10:00 a.m.

1. Use the following structure and data definitions to complete this question.

```
(define-struct manager (name company branch))
;; A Manager is a (make-manager Str Str Sym)
;;
(define-struct supervisor (name company branch))
;; A Supervisor is a (make-supervisor Str Str Sym)
;;
;; A Staff is one of:
;; * a Manager
;; * a Supervisor
```

   (a) Write the template function for `Staff`.
   (b) Write a function `same-branch?` that consumes two `Staff` structures and produces true if both are working in the same branch and the same company, and false otherwise. For example:

```
(define man1 (make-manager "Joe Clarke" "Company 1" 'finance))
(define sup1 (make-supervisor "Trish Lego" "Company 1" 'finance))
(define man2 (make-manager "Bill Lewis" "Company 1" 'sales))

(same-branch? man1 sup1) => true
(same-branch? sup1 man2) => false
```

2. Use the following structure and data definitions to complete this question.

```
(define-struct triangle (p1 p2 p3))
;; A Triangle is a (make-triangle Posn Posn Posn)
```

   A triangle's three vertices are each represented by a `Posn` structure. A triangle is invalid when two of its three vertices are the same, when the area of the triangle is zero, or when all three vertices of the triangle are collinear, forming a straight line. The Shoelace formula may be used to calculate the area of a triangle with vertices (x_1, y_1), (x_2, y_2), and (x_3, y_3) as follows:

$$Area\ of\ a\ triangle = \frac{1}{2}|\ x_1(y_2 - y_3)\ +\ x_2(y_3 - y_1)\ +\ x_3(y_1 - y_2)|$$

   Write a Racket function `valid-triangle?` that consumes a `Triangle` structure and produces true if the triangle is valid and false otherwise. For example:

```
(define p1 (make-posn 2 3))
(define p2 (make-posn 8 3))
(define p3 (make-posn 8 12))
(define p4 (make-posn 2 3))

(valid-triangle? (make-triangle p1 p2 p3)) => true
(valid-triangle? (make-triangle p1 p2 p4)) => false
```

   You may read more about the Shoelace formula at https://en.wikipedia.org/wiki/Shoelace_formula.

3. Use the following structure and data definitions to complete this question.

```
(define-struct midterm (mark portion))
;; A Midterm is a (make-midterm Num Num)
;; requires: 0 <= mark <= 100
;;           0 <= portion <= 100
;;
(define-struct result (student course mark))
;; A Result is a (make-result Str Sym Num)
;; requires: 0 <= mark <= 100
```

Write a Racket function `get-result` that consumes a student name string, a course code symbol, a `Midterm` structure and the final exam mark, and produces the result, a `Result` structure. The final mark is a number between 0 and 100, inclusive. The mark in an exam result is calculated by adding the product of mark and portion in midterm with the product of final and (1 – midterm portion).

For example:

```
(define m1 (make-midterm 75 0.50))
(define m2 (make-midterm 90 0.35))

(get-result "Joe Andy" 'CS115 m1 80.5) =>
        (make-result "Joe Andy" 'CS115 77.75)

(get-result "Jane Boole" 'CS115 m2 65.2) =>
        (make-result "Jane Boole" 'CS115 73.88)
```