# Assignment 3
## Due at 9:00 AM on Wednesday, October 17

For this and all subsequent assignments, you are expected to use the design recipe when writing functions from scratch, including helper functions. Do not copy the purpose directly from the assignment description. The purpose should be written in your own words and include reference to parameter names of your functions.

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 e-mail to your instructors or tutors. They will be not accepted by course staff as assignment submissions.

Test data for all questions will always meet the stated assumptions for consumed values.

Please read the course Web page for more information on assignment policies and how to organize and submit your work.

Be sure to download the interface file from the course Web page and to follow all the instructions listed in the Style Guide (on the Web page and in the printed package of handouts). Specifically, your solutions should be placed in files `a03qY.rkt`, where `Y` is a value from 1 to 4.

For full marks, it is not sufficient to have a correct program. Be sure to follow all the steps of the design recipe, including the definition of constants and helper functions where appropriate.

**Contracts:** You are expected to use both built-in and user-defined types. See Section 1.6 of the Style and Submission Guide for details.

**Language level:** Beginning Student.

**Coverage:** Module 4. Use only material from Modules 1–4 in your solutions.

**Useful structure and data definitions:**

(*define-struct cal* (*month day*))
;; A **cal** is a structure (*make-cal m d*) where
;; *m* is an integer in the range 1-12
;; *d* is an integer in the range 1-31
;; and *m* and *d* represent a calendar entry in a non-leap year.

(*define-struct interval* (*start end*))
;; An **interval** is a structure (*make-interval s e*) where
;; *s* and *e* are both *cal* structures,
;; and *s* is no later than *e* in the same year.

(*define-struct festival* (*name location cost dates*))
;; A **festival** is a structure (*make-festival n l c d*) where
;; *n* and *l* are both strings,
;; *c* is the symbol 'free or 'charge, and
;; *d* is an interval.

**Constant definitions used in the examples below:**

(**define** *cal3-1* (*make-cal* 3 1))
(**define** *cal3-4* (*make-cal* 3 4))
(**define** *cal3-8* (*make-cal* 3 8))
(**define** *cal3-10* (*make-cal* 3 10))
(**define** *cal8-9* (*make-cal* 8 9))
(**define** *cal8-12* (*make-cal* 8 12))
(**define** *cal8-23* (*make-cal* 8 23))
(**define** *cal8-26* (*make-cal* 8 26))

(**define** *reelsdates* (*make-interval cal3-1 cal3-4*))
(**define** *comedydates* (*make-interval cal3-8 cal3-10*))
(**define** *bluesdates* (*make-interval cal8-9 cal8-12*))
(**define** *buskersdates* (*make-interval cal8-23 cal8-26*))

(**define** *reels* (*make-festival* "Rainbow Reels" "Waterloo" 'charge *reelsdates*))
(**define** *comedy* (*make-festival* "Uptown Waterloo Comedy Festival" "Waterloo" 'charge *comedydates*))
(**define** *blues* (*make-festival* "Kitchener Blues Festival" "Kitchener" 'free *bluesdates*))
(**define** *buskers* (*make-festival* "Busker Carnival" "Waterloo" 'free *buskersdates*))

1. Create a function *safe-make-cal* that consumes two integers, *m* and *d*, and either produces the *cal* with month *m* and *d*, if *m* and *d* satisfy the necessary requirements in the definition, or 'impossible if there is no calendar entry with month *m* and day *d* in a non-leap year calendar. For example, (*safe-make-cal* −3 4) will produce 'impossible and (*safe-make-cal* 2 5) will produce (*make-cal* 2 5).

2. Create a function *overlap?* that consumes two *interval* structures, *int1* and *int2*, and produces *true* if there are any dates included in both intervals and *false* otherwise. Note: an interval is considered to include both the starting and ending dates. For example, (*overlap? reelsdates comedydates*) will produce *false* and (*overlap? reelsdates* (*make-interval cal3-4 cal3-8*)) will produce *true*.

3. Create a function *festival-season* that consumes a *festival* structure *fest* and produces 'fall, 'winter, 'spring, or 'summer depending on the month in which the *fest* starts ('fall for September through November, 'winter for December through February, 'spring for March through May, and 'summer for June through August). For example, (*festival-season blues*) will produce 'summer.

4. Create a function *better-festival* that consumes two *festival* structures, *fest1* and *fest2*, and produces the name of the better festival, or the name of fest1 if they are equally good. The order of festivals from best to worst is as follows: free and in Waterloo, free but not in Waterloo, in Waterloo but not free, anything else. A festival is in Waterloo if the location is exactly the string "Waterloo". For example, (*better-festival blues buskers*) will produce "Busker Carnival" and (*better-festival reels comedy*) will produce "Rainbow Reels".