- Do **not** use any conditional expression (`cond`) on this assignment
- For this and all subsequent assignments, you are expected to use the design recipe when writing functions from scratch, including helper functions.
- **For full marks, it is not sufficient to have a correct program. Be sure to follow all the steps of the design recipe. Read the Style Guide carefully to ensure that you are following the proper conventions. *In addition, your solution must include the definition of constants and helper functions where appropriate*.**
- 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.
- 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.
- 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.
- You may post general assignment questions using the discussion groups on Waterloo LEARN. Choose Connect then 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.
- 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 `a2qY.rkt`, where `Y` is a value from `1` to `3`.
- Since some files you submit will contain more than one function, **it is very important that your code runs**. If your code does not run then none of the functions can be tested for correctness.

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 assignment and a 5% reduction of 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.
- Be sure to read the Plagiarism section at:
  https://www.student.cs.uwaterloo.ca/ cs115/

Language level: Beginning Student
Coverage: Modules 1 and 2

Refer to the String documentation found at: https://www.student.cs.uwaterloo.ca/ cs115/resources/strings.pdf for helpful Racket functions.

Make sure to use constants and helper functions whenever appropriate.

1. A group of high school students and staff had a day trip. The school principal wants to compute the total cost of the trip.

   - Transportation costs \$35 for a staff member and \$25 for a student.
   - A ticket to the museum costs \$21.5 and a student gets 25% discount.
   - All-you-can-eat restaurant costs \$22.75 per person (including soft drinks).

   Write a Racket function `trip-cost` that consumes two natural numbers `nstaff` and `nstudents`, representing the number of staff members and students who participated in the trip, and produces the total cost of the trip.
   For example:

   - `(trip-cost 4 35) => 2552.625`
   - `(trip-cost 1 4) => 334.75`

2. Write a Racket function `middle` that consumes three different integers `a`, `b`, and `c` and produces the middle value.
   For example:

   - `(middle 11 5 10) => 10`
   - `(middle -4 0 -10) => -4`

3. Write a Racket function `reverse-int` that consumes four-digit natural number, `n`, and produces the same consumed number in reverse order of digits without leading zero (A leading zero is any 0 digit that comes before the first nonzero digit in a number, for example 007 has two leading zeros).
   For example:

   - `(reverse-int 3206) => 6023`
   - `(reverse-int 1500) => 51`

   Hint: you may need to use the `string->number`, `string-ith`, and/or `number->string` functions.

4. For this question, you will perform step-by-step evaluations of Racket programs, by applying substitution rules until you either arrive at a final value or you cannot continue. You will use an online evaluation tool that we have created for this purpose. To begin, visit this webpage:

   `https://www.student.cs.uwaterloo.ca/~cs115/stepping`

   Read the notes on the following page carefully before you attempt to solve this problem.

Notes:

- The use of https is important; that is, the system will not work if you omit the s. This link can also be found on the CS115 course webpage, under the Stepping heading.
- You will need to authenticate yourself using your Quest/WatIAm ID and password.
- Once you are logged in, try the "Warmup questions" under "CS 115 Assignment 2", in order to get used to the system.
- You can re-enter a step as many times as necessary until you get it right, so keep trying until you completely finish every question.
- All you have to do is complete the questions online, we will be recording your answers as you go, and there is no file to submit.
- Note however that you are not done with a question until you see the message Question complete!
- You should see this once you have arrived at a final value and clicked on "simplest form" (or "Error", depending on the question).
- You should **not** use DrRacket's Stepper to help you with this question for several reasons.
  - First, as mentioned in class, DrRacket's evaluation rules are slightly different from the ones presented in class, but we need you to use the evaluation rules presented in class.
  - Second, in an exam situation, you will not have DrRacket's Stepper to help you, and there will definitely be step-by-step evaluation questions on at least one of the exams.