Lab 03: The design recipe and helper functions

The purpose of this lab is for you to get practice in using the design recipe, including testing. Make sure to use check-expect or check-within, as appropriate. We have provided public tests that you can use to check your work, but please do not use them as a replacement for designing your own.

Create a separate file for each question. Keep them in your “Labs” folder, with the name liiiqj for Lab ii, Question j.

Download the headers for each function from the file labinterface03.rkt linked off the “Labs” page on the course Web site.

After you have completed a question (except class exercises), including creating tests for it, you can obtain feedback by submitting it and requesting a public test. Follow the instructions given in the Style Guide.

Language level: Beginning Student.

1. [Class exercise with lab instructor assistance] Create a function ring-volume that consumes the inner radius, outer radius, and thickness of a ring and produces its volume.

2. Create a function first-char that consumes a nonempty string and produces a string consisting of the first character in the original string. Do not use string-ref.

3. Create a function last-char that consumes a nonempty string and produces a string consisting of the last character in the original string. Do not use string-ref.

4. Create a function shipping-bill that determines the cost for shipping merchandise. It consumes the handling charge (a fixed cost for a shipment of any size), the charge per kilogram, the weight of a box in kilograms, and the number of identical boxes.

5. The airport parking lot has rates by the week and by the day, where you pay by the weekly rate of $74.95 for each complete week (any consecutive seven days) and by the daily rate of $14.95 for any remaining days. Create a function airport-parking that consumes an integer number of days and produces the bill. Be sure to use constants where appropriate. Note: don’t worry if you produce a number that looks like 5 instead of 5.00 or a number that looks like 5.1 instead of 5.10.

6. Films are composed of many film clips that are joined together by a film editor. Create a function, film-choice, that produces a string indicating the costs of creating a film based on two pricing options. The function film-choice consumes at total of six values: the number of clips, the length of the film, and two pairs of values that describe the pricing for each of two options. The pricing for each option is determined by a breakpoint: the number of clips that the editor works on at the full rate of $100 each, and a discount rate: a percentage rate reduction for the clips that the editor works on past the breakpoint. For each option, the total cost is the sum of the charge for editing the clips and a charge based on the length of the film which is $100 per minute. It is possible for the breakpoints in the pricing options to be greater than the number of clips in the movie.

For example, suppose the number of clips is 10, the length of the film is 30 minutes, the first option has a breakpoint of 5 and a discount of .25, and the second option has a breakpoint of 8 and discount of 0.5. Then the total cost for the first option will be the charge based on the length of the film, which
is \((30 \cdot 100)\), plus the cost for 5 clips at $100 each and the cost for 5 clips at $75 each. This is a total of 
$3000 + 500 + $375 = $3875. The total of $3900 for the second option would be calculated in a
similar fashion. The string produced by \textit{film-choice} for option totals of $3875 and $3900 will be:

"Cost for option 1 is 3875 and cost for option 2 is 3900"

You may find it convenient to use the function \texttt{number->string}. \textbf{Your solution should be making}
use of several helper functions.

7. Optional open-ended questions

(a) Create a function that consumes a string (an adjective) and produces a comparative by
adding “er” (or adding “more” to the front) or into a superlative by adding “est” (or
adding “most” to the front).

(b) Pig Latin is a (not very difficult to decode) scrambling of English words created by
taking the initial consonant sound off the front of the word and moving it to the end
and then appending the string “ay”. For example, “apple”, “banana”, and “grape” are
translated to “appleye” (no consonant sound at the front), “ananabay” (single letter
consonant sound), and “apegray” (multiple-letter consonant sound). Create a function
that converts a string into a simple version of Pig Latin by just adding “ay” to the end
of the string. Now create a function that moves the first letter in the string (whether or
not it is a consonant) to the end and then appends “ay”. We will refine this function as
we learn more.

(c) If time permits, consider creating a function that consumes a string and a position
number, and returns the string formed by removing the letter in that position. Or a
function that consumes a string and two position numbers, and returns the string formed
by removing all the letters in between the two positions.

\textbf{Helpful tips}

\textbf{Commenting and uncommenting} Select a block of text (part of a line, one line, or more) in the
Definitions window. Under “Racket” on the menu bar, select “Comment Out with Semi-
colons”. You can undo the change using “Uncomment”. \textit{Do not use comment boxes, or your}
assignments will be unmarkable.

\textbf{Indenting} Select a block of text in the Definitions window. Under “Racket” on the menu bar,
select “Reindent”. You can also use “Reindent All” to reindent the entire program. To indent
a single line, put the cursor on the line and press tab.

\textbf{Jumping to a definition} In the top left corner of your window is an arrow labelled “(define ...)”. When you click here, you get a menu of all your definitions. You can choose whether they should be sorted by their order in the file or alphabetically.