Goals of this tutorial

You should be able to...

- write the full design recipe for simple arithmetic functions.
- understand when and how to use constants.

Announcements

- MarkUs Basic tests:
  - Are set up for every assignment
  - Do not thoroughly test your code
  - Ensure we can run more thorough tests on your code after the due date
  - The results are automatically emailed to your uWaterloo email. You can also check the results on Markus.
  - Are not related to the tests in your code
Review: The five design recipe components

Purpose: Describes what the function produces. You should meaningfully include parameter names in your purpose statement.

Contract: Describes what type of arguments the function consumes and what type of value it produces.

Additional contract requirements: If there are important constraints on the parameters that are not fully described in the contract, add an additional requires section to “extend” the contract.

Examples: Illustrate the use of the function.

Definition: The Racket definition (header and body) of the function.

Tests: A thorough set of function arguments and expected function values.

Clicker Question - Design Recipe
For the built-in function expt, which of the following has the best purpose, contract and example?

A
;; (expt arg1 arg2) raises arg1 to the power of arg2
;; expt: num num → num
;; Example:
(check-expect (expt 3 2) 9)

B
;; (expt arg1 arg2) raises arg1 to the power of arg2
;; expt: Num Num → Num
;; Example:
(check-expect (expt 3 2) 9)

C
;; expt: Nat Nat → Nat
;; Purpose: To raise a natural number to a power
;; Example:
(check-expect (expt 4 2) 16)

D
;; (expt arg1 arg2) raises arg1 to the power of arg2
;; expt: num num → num
;; Example:
(check-expect (expt 2 3) 8)

E
;; (expt arg1 arg2) raises an integer to a power
;; expt: Int Int → Int
;; Example:
(check-expect (expt 3 2) 9)

How to find the help pages

• DO NOT use Google search. It will land you at the wrong language level, typically at the full Racket help page.

• Open DrRacket: Help menu > Help Desk (this opens a browser window) > Teaching > How to Design Programs Languages > Select the appropriate language level (e.g. Beginning Student).

• NOTE the categorized list of functions on the left side bar.

• If you must Google, then you have to add the teaching language name to your query, e.g. "racket beginning student".
Purpose & Contract
Use the help desk to create a purpose and contract for even?.

Purpose & Contract
Use the help desk create to a purpose and contract for modulo.

Clicker Question - Racket Errors
Which of the following is an error-free Racket expression in “Beginning Student”?

A (3 * 5 + 1)
B (/ 5 (− 1 1))
C (+ (+ 2 3 −4) 9)
D (* (3 5) (3))
E They all have some kind of error.
Group Problem - direct translation

This is the function that calculates the area of a trapezoid:

\[
\text{area-of-trapezoid}(\text{base1}, \text{base2}, \text{height}) = \frac{1}{2} \cdot (\text{base1} + \text{base2}) \cdot \text{height}
\]

Translate this function into Racket, providing the full design recipe.

Review: Advantages of constants

- Can give meaningful names to useful values (e.g. interest-rate, passing-grade, and speed-of-light).
- Reduces typing and errors when such values need to be changed.
- Makes programs easier to understand.
- Constants can be used in any expression, including the body of function definitions.
- Sometimes called variables, but their values cannot be changed (until CS 136).

Group Problem - weighted-assn-average

Congratulations! You are now enrolled in CS 987. Here is the grade breakdown:

- Assignment 1 - 15%
- Assignment 2 - 10%
- Assignment 3 - 5%
- Assignment 4 - 20%
- Dance Battle - 50%

To pass the course, you must pass the weighted assignment average.
Group Problem - weighted-assn-average

Write a Racket function, `weighted-assn-average`, which consumes 4 numbers which represent grades for Assignment 1, Assignment 2, Assignment 3 and Assignment 4 (in that order). This function should produce the weighted assignment average in the course (as a percentage in the range 0 to 100, but not necessarily an integer). Include a purpose and contract. You may assume that all the arguments are integers between 0 and 100 inclusive.