CS 111 - Week 5 Lab Exercise - 2016-09-23

Deadline

Due by the end of lab today. (Submit whatever you have by the end of lab, even if incomplete.)

How to submit

Submit your Definitions window (`.rkt` file) for this lab using `~st10/111submit` on nrs-labs, with a homework number of `85`.

Purpose

To practice using Racket lists.

Important notes

- You are required to work in **pairs** on this lab exercise If you are not pair-programming, then you may not receive full credit for your lab exercise.
- If you have a question during lab, and I am helping another pair, add one or both of your names to the "Next:" list on the board, and I will get to you as soon as I can.
- There is a posted file of list-related reminders, `111lab05-reminders.rkt`, available from the CS 111 public course web site, under "In-class Examples".
  - Hint: in that file, you can obtain copies of some of the templates you are asked to paste into YOUR lab file below!

Problem 1

The main purpose of this problem is to provide you with some practice writing expressions involving lists. You are NOT writing any new functions for this problem!

1 part a

Create a new Racket Definitions window `lab5.rkt`, and put BOTH YOUR NAMES in a comment within it. Then:

- **Paste or type in** the comment containing the data definition for a Racket list. (It is OK if you do not also copy over the data definition for Anything...)
- **Paste or type in** the comment containing the TEMPLATE for a function that "walks through" a list
- Decide on a theme/topic, and define a named constant, with an appropriate, descriptive name, whose value is a list of at least FOUR things related to that theme/topic.
  - For this part, list abbreviations will **not** be accepted.

1 part b

Now, **USING your named constant list**:

- Write an expression whose value is the **first** thing in your named constant list.
- Write an expression whose value is the list of **all BUT the first thing** in your named constant list.
- Write an expression whose value is **JUST** the **SECOND** thing in your named constant list.
- Write an expression whose value is **JUST** the **THIRD** thing in your named constant list.

1 part c

Note that Racket does have a built-in `length` function for lists -- (we designed `len` because it is such a good first example of a function that "walks through" a list... 8 -)):

```
; signature: length: list -> number

; purpose: expects a list, and produces the number of (top-level)
;    elements in that list

(check-expect (length (cons 1 (cons 8 (cons 27 empty))))
              3)
```

USING your named constant list:

- Write an expression whose value is the length of your named constant list.
- Write an expression whose value is the length of the list of all BUT the first thing in your named constant list.

Problem 2

2 part a

We want to work with some lists of numbers. So, paste or type in the data definition comment for a `NumberList`, and then paste or type in the comment containing the TEMPLATE for a function that "walks through" a `NumberList`.

2 part b

Following the design recipe, design and write a function that expects a list of numbers, and just returns the **sum** of the numbers in that list. That is, if this function has as its argument a list of numbers containing 7, 10, and 47, then the value produced would be 64. (And, if called with the `empty` list as its argument, this function should produce the value 0.)