Carleton University
Department of Systems and Computer Engineering
SYSC 3101 - Programming Languages - Winter 2023

Lab 1 - Introduction to Racket (Scheme)

References

Two documents at the Racket website provide plenty of information about the Racket dialect of Scheme:

The Racket Guide, https://docs.racket-lang.org/guide/index.html

The Racket Reference, https://docs.racket-lang.org/reference/index.html

A guide to the DrRacket IDE can be found here:

http://docs.racket-lang.org/drracket/index.html

To download DrRacket IDE

https://download.racket-lang.org/

Racket Coding Conventions

Indentation: DrRacket provides a command to reformat code in the definitions area (Menus >Racket > Reindent All).

Procedure names: A common convention for choosing a procedure name is to use a noun or noun-phrase that describes what the procedure returns. This makes expressions that apply the procedure easier to read. For example, use `discounted-price` instead of `calculate-discounted-price`.

Predicate names: Predicates are procedures that return boolean values (true or false). The names of these procedures should end in `?`; for example, `odd?` or `good-enough?`.

Adding comments: A ; starts a line comment.

Getting Started

Launch the DrRacket IDE.

If necessary, configure DrRacket so that the programming language is Racket. To do this, select Language > Choose Language from the menu bar, then select The Racket Language in the Choose Language dialog box.

`#lang racket` should appear at the top of the definitions area. Don't delete this line.

Do not use special forms that have not been presented in lectures. Specifically,

- Do not use `set!` to perform assignment; i.e., rebind a name to a new value.
- Do not use `let` expressions to create local variables.
- Do not use `begin` expressions to group expressions that are to be evaluated in sequence.

Exercise 1 (5 mark)

The local supermarket needs a program that can compute the value of a bag of coins. Define procedure `sum-coins`. It consumes four numbers: the number of pennies, nickels, dimes, and quarters in the bag. Its result is the amount of money in cents.

Test your procedure by typing these expressions in the interactions window:

```
(sum-coins 1 0 0 0)    ; result should be 1
(sum-coins 0 1 0 0)    ; result should be 5
(sum-coins 0 0 1 0)    ; result should be 10
(sum-coins 0 0 0 1)    ; result should be 25
(sum-coins 1 1 1 1)    ; result should be 41
```

Exercise 2(5 marks)

Define procedure `interest`. It consumes a bank account balance (the amount of money in the account), and calculates the amount of interest that the money earns in one year. The bank pays a flat 4% for balances up to $1,000, a flat 4.5% per year for balances above $1000 and up to $5,000, and a flat 5% for balances of more than $5,000.

Test your procedure by typing these expressions:

```
(interest 500)   ; result should be 20
(interest 1000)  ; result should be 40
(interest 2000)  ; result should be 90
(interest 5000)  ; result should be 225
(interest 10000) ; result should be 500
```

Exercise 3 (5 marks)

Define procedure `balance`. It consumes a bank account balance (the amount of money in the account), and new balance after adding the annual amount of interest according to the rules in Exercise 2.

Test your procedure by typing these expressions:

```
(balance 500)   ; result should be 520
(balance 1000)  ; result should be 1040
(balance 2000)  ; result should be 2090
(balance 5000)  ; result should be 5225
(balance 10000) ; result should be 10500
```

Exercise 4 (5 marks)

Define procedure **variable_Interest** that calculates the amount of interest that the money earns in one year. The bank pays a 4% per year for the first $1,000, a 4.5% for the portion above $1,000 and up to $5,000, and a 5% for the portion above $5,000. (Note: ***Don't use Recursive Procedures***)

Test your procedure by typing these expressions:

```
(variable_Interest   500)   ; result should be 20

(variable_Interest   1000)  ; result should be 40

(variable_Interest   2000)  ; result should be 85

(variable_Interest   5000)  ; result should be 220

(variable_Interest   10000) ; result should be 470
```

Remember to demonstrate your work to TA to get the lab grade then submit your code as PDF file on Brightspace.