Ann Liberman 3/26/2021
CISC 3140

Lab #7

I. **The version of Scheme I used was `Guile`, via the school server.**

II. **The document I used to reference the specifications:**
https://www.gnu.org/software/guile/manual/html_node/Scripting-Examples.html

III. **Description of the problem**: The goal is to convert weather units from celsius to fahrenheit and back.

- **The variables are**: The input variable are tempF & tempC
- **The inputs are**:

For F → C
- 100 degrees
- 32 degrees
- 0 degrees, and
- 56 degrees

For C → F
- 10 degrees
- 15 degrees
- 40 degrees, and
- 5 degrees

- **The outputs are:**

For F → C
- 340 / 9, or 37.7 degrees
- 0 degrees
- -160 / 9, or -17.7 degrees
- 40 / 3, or 13.3 degrees

For C → F
- 50 degrees
- 59 degrees
- 104 degrees
- 41 degrees

IV. **Think about what the solution will look like. Explain this solution:**

- In order to solve this problem, first I defined a values for x, in the first case, I did:
  - (define x 100)
- Then I used the formula to convert from F to C, which is to subtract 32 from x, then divide the result by (9/5)
  - (/ (- x 32) (/ 9 5))

- Then I changed the values of x to reflect what I wanted my input to be (in this case; 100 degrees, 32 degrees, 0 degrees, and 56 degrees).
- I checked the outputs against a calculator to make sure they were correct, and they were! I also reversed the process for converting from C to F

V.

```
;; ftoc : number → number
;; converts a Fahrenheit temp to a Celsius temp
(define (ftoc tempF)
(* (/ 5 9) ( tempF 32)))

;; listftoc : (listof number) → (listof number)
;; converts a list of Farenheit temps to a list of Celsius temps
(define (listftoc list-Ftemps)
  (cond
    [(empty? list-Ftemps) empty]
    [(cons? list-Ftemps) (cons (ftoc (first list-Ftemps))
                               (listftoc (rest list-Ftemps)))]))

;; ctof : number → number
;; converts a Celsius temp to a Fahrenheit temp
(define (ctof tempC)
(+ (* tempC (/ 9 5)) 32)

;; listctof : (listof number) → (listof number)
;; converts a list of Celsius temps to a list of Farenheit temps
(define (listctof list-Ctemps)
  (cond
    [(empty? list-Ctemps) empty]
    [(cons? list-Ctemps) (cons (ctof (first list-Ctemps))
                               (listctof (rest list-Ctemps)))]))
```

VI. **Scheme log for direct in-terminal use:**

```
scheme@(guile-user) [4]> (define x 100)
scheme@(guile-user) [4]> (/ (- x 32) (/ 9 5) )
$6 = 340/9
scheme@(guile-user) [4]> (rationalize (/ (- x 32) (/ 9 5) ) )
```

```
;;; <stdin>:14:0: warning: possibly wrong number of arguments to `rationalize'
ERROR: In procedure rationalize:
ERROR: Wrong number of arguments to #<procedure rationalize (_ _)>

Entering a new prompt.  Type `,bt' for a backtrace or `,q' to continue.
scheme@(guile-user) [5]> ,bt
           0 (rationalize 340/9)
scheme@(guile-user) [5]> (real (/ (- x 32) (/ 9 5) ) )
;;; <stdin>:16:0: warning: possibly unbound variable `real'
<unnamed port>:16:0: In procedure #<procedure 2530580 at <current input>:16:0 ()>:
<unnamed port>:16:0: In procedure module-lookup: Unbound variable: real

Entering a new prompt.  Type `,bt' for a backtrace or `,q' to continue.
scheme@(guile-user) [6]> (scm_to_double (/ (- x 32) (/ 9 5) ) )
;;; <stdin>:17:0: warning: possibly unbound variable `scm_to_double'
<unnamed port>:17:0: In procedure #<procedure 229b540 at <current input>:17:0 ()>:
<unnamed port>:17:0: In procedure module-lookup: Unbound variable: scm_to_double

Entering a new prompt.  Type `,bt' for a backtrace or `,q' to continue.
scheme@(guile-user) [7]> (let x 0)
While compiling expression:
ERROR: Syntax error:
unknown file:18:0: let: bad let in form (let x 0)
scheme@(guile-user) [7]> (define x 0)
scheme@(guile-user) [7]> (/ (- x 32) (/ 9 5) )
$7 = -160/9
scheme@(guile-user) [7]> (define x 32)
scheme@(guile-user) [7]> ^[[A
;;; <unknown-location>: warning: possibly unbound variable `#{\x1b;}#'
ERROR: In procedure #<procedure 24bd6a0 ()>:
ERROR: In procedure module-lookup: Unbound variable: #{\x1b;}#
Entering a new prompt.  Type `,bt' for a backtrace or `,q' to continue.
While reading expression:
ERROR: In procedure scm_i_lreadparen: #<unknown port>:32:1: end of file
scheme@(guile-user) [8]> (/ (- x 32) (/ 9 5) )
$8 = 0
scheme@(guile-user) [8]> (define x 56)
scheme@(guile-user) [8]> (/ (- x 32) (/ 9 5 ) )
$9 = 40/3
scheme@(guile-user) [8]> (*(+ x 32) (/ 9 5) )
$10 = 792/5
scheme@(guile-user) [8]> (define x 10)
scheme@(guile-user) [8]> (* (+ x 32) (/ 9 5) )
$11 = 378/5
scheme@(guile-user) [8]> (+ (* x (/ 9 5)))
$12 = 18
scheme@(guile-user) [8]> (+ (* x (/ 9 5)) 32)
$13 = 50
```

```
scheme@(guile-user) [8]> (define x 15)
scheme@(guile-user) [8]> (+ (* x (/ 9 5)) 32)
$14 = 59
scheme@(guile-user) [8]> (define x 40)
scheme@(guile-user) [8]> (+ (* x (/ 9 5)) 32)
$15 = 104
scheme@(guile-user) [8]> (define x 5)
scheme@(guile-user) [8]> (+ (* x (/9 5)0 32)
^[[D
^[[A^[[A^[[A`quit

While reading expression:
ERROR: In procedure scm_i_lreadparen: #<unknown port>:49:1: end of file
scheme@(guile-user) [8]> (define x 5)
scheme@(guile-user) [8]> (+ (* x (/9 5)) 32)
;;; <stdin>:50:8: warning: possibly unbound variable `/9'
<unnamed port>:50:8: In procedure #<procedure 2714700 at <current input>:50:0 ()>:
<unnamed port>:50:8: In procedure module-lookup: Unbound variable: /9

Entering a new prompt.  Type `,bt' for a backtrace or `,q' to continue.
scheme@(guile-user) [9]> (+ (* x (/ 9 5)) 32)
$16 = 41
scheme@(guile-user) [9]>
```