CS 1101 - A-term 16

Homework 0 - Composing Functions

Due: Wednesday, August 31 at 5pm

You will be doing this homework *individually*.

Assignment Goals

- To become familiar with the terms *expression, operator, operand, function, constant, function definition, function call, argument, parameter*.
- To understand how Racket evaluates expressions.
- To make sure you can write simple functions.
- To learn how to document functions with a signature and purpose.
- To become familiar with DrRacket's helpdesk.
- To be able to combine simple functions into more complicated functions.
- To gain familiarity with the use of images as data.

The Assignment

Preliminaries

0. Start up DrRacket. If this is the first time you've brought up DrRacket, you will be asked to choose a language. From the **Language** menu, choose *Choose Language*. From *How to Design Programs*, select *Beginning Student*. If a *Show Details* button appears in the bottom left-hand corner of the window, click it. On the right-hand side of the window look at the panel that says *Output Syntax*. Under *Constant Style* choose *true false empty*. Click the *OK* button. Now, in the definitions window, write a comment that includes your name and your username. From the **File** menu, choose *Save Definitions As*.... Name your file *yourLastName-hw0*, where *yourLastName* is your surname. Click the **Run** button.

   NOTE: Your file will be saved as *yourLastName-hw0.rkt*. DrRacket saves the work done in the Definitions window only.

   As you answer each of the remaining problems, use comments in your file to clearly indicate the number of the problem you are solving, e.g..

```
;;
;;  Problem 1
;;
```

Expression Evaluation

1. Evaluate each of the following expressions. Show every step in the evaluation. Provide your answers as comments in the Definitions window. For example, to evaluate the expression

```
(* (+ 4 3) 9)
```

   You should provide the following comments in the Definitions window:

```
;; (* (+ 4 3) 9)
;; (* 7 9)
;; 63
```

- (a.)

  ```
  (if (< (string-length (string-append "CS" "1101"))
         (* (- 8 5) (/ 30 15)))
      36
      (sqrt (+ (sqr 3) (- 11 4))))
  ```

- (b.)

  ```
  (and (> (+ (double (+ 1 3)) 10) (+ 15 5)) (> (double (+ 2 3)) 4))
  ```

  where `double` is defined as

  ```
  (define (double n)
     (* n 2))
  ```

Composing Functions

Provide a signature and purpose for each function you write.

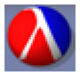
2. In class and in the videos you were introduced to some built-in functions (like `rectangle` and `text`) from the *image* library. Add the following line to the Definitions window:

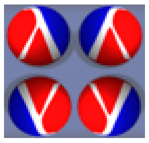
   ```
   (require 2htdp/image)
   ```

   Now click the **Run** button. The functions in the image library are now available to your program.

   From the *Help* menu, choose *Help Desk*. A browser window will open. Under the *Teaching* heading, choose *How to Design ProgramsTeachpacks*. In the left margin, click on *HtDP/2e Teachpacks*. Here you will find a description of all the pre-defined functions in the *image.ss* library. Notice the ...*search manuals*... box at the top of the page. Type in `image-width` to find information about that function. When the DrRacket Helpdesk provides the matches for `image-width`, choose the one from "2htdp/image".

   Develop a function called *four-square* that consumes an image and produces an image that consists of 4 copies of the original image, as shown in this example:

   if the original image looks like this: 

   then the produced image should look like this: 

   Use the DrRacket help desk to learn about the following functions (available in the 2htdp/image library) that you might consider using for this problem:

   - flip-vertical
   - flip-horizontal
   - beside
   - above

   (You're not restricted to using these functions. You may solve the problem any way you want as long as the image is duplicated in the same manner as the image in the illustration.)

Google images is a good repository for images. Use a square-shaped image when you try out your program. To copy an image, right-click on the image, select Copy, then in the DrRacket Definitions window, select Paste. You should use *define* to name any images you copy into your program.

Note: Make sure you provide a signature and purpose for your function. You do not have to test this problem with `check-expect`.

3. A local performing arts center books various events. Every event costs the center $1500 (for utility costs, custodial help, etc.) plus $2 per attendee (for program printing, etc.). You are to develop a set of functions that will calculate the profit for an event, based on the number of attendees and the cost of a ticket.
   - (a) develop a function that consumes the number of attendees at an event and produces the cost to the performing arts center to hold the event. Name your function `event-cost`.
   - (b) develop a function that consumes the number of attendees and the price per ticket and produces the income generated by ticket sales. Name your function `ticket-revenue`.
   - (c) develop a function that consumes the number of attendees and the price per ticket and produces the profit for the event. Use the functions you developed in parts (a) and (b). Name your function `event-profit`.

   Use defined constants where appropriate. Make sure each function you define is documented with a signature and a purpose. Each function should be tested using `check-expect`. *NOTE: You must name each function with the exact name specified in the problem. Your signature must conform to the problem description. Otherwise, we won't be able to run our automated tester on your program, and you'll lose points*. Programs that don't work with our auto-tester (and thus must be tested manually) will be penalized with a deduction of 10% of the total number of points for the assignment.

Grading

Here is the grading rubric the graders will use for Homework 0.

What to Turn In

Using InstructAssist, choose `Tools -> File Submission` to turn in the file (*yourLastName-hw0.rkt*) containing your Definitions Window content. Your name and your wpi ccc username must be listed in a comment at the beginning of your file. Programs submitted after 5pm on Wednesday, August 31 will be considered late. No submissions will be accepted after 5pm on Thursday, September 1.
