1. Final Exam

This is the final exam for Intermediate Computer Science, Semester I, 2017-2018. Except as noted below, you may not use materials previously written by you or anyone else.

There are two sections: written (1/3) and programming (2/3). Before you use a computer, you must turn in your answers to the written questions.

Write all of your solutions on a separate sheet of paper.

Written Questions

(30 points: do only 3)

In the written portion, you may not consult any sources. Suggested time: 30 minutes.

1. (10 pts) What is one signifiant benefit of writing the signature of a function? Include a realistic example that supports the benefit you have chosen.
2. (10 pts) Shelly is working with a mouse-handler:

   ```
   ; Purpose: return a posn of last mouse position
   (define (mouse-h m e x y)
       (make-posn x y))
   ```

   1. Critique Shelly's choice of a signature:

      ```
      mouse-h: model string number number -> model
      ```

   2. Revise to make a complete, correct `mouse-h` signature and function.
3. (10 pts) Usually a function signature includes `number` or `string`. Sometimes it is appropriate to use `any` as part of a signature.
   1. Give an example of a function whose signature should include `any`.
   2. Explain why the `any` parameter is useful in this case.
4. (10 pts) Analyze the function below.
   1. Write one sentence explaining what `roll-dice` does.
   2. Write a test or edit so the function is testable.

      ```
      (define dice1 (random 10))
      (define dice2 (random 10))
      ; purpose: sum two random ten sided dice rolls
      ; each die can roll 1 through 10
      (define (roll-dice n)
        (+ (dice1) (dice2)))
      ```

Programming Questions

(60 points: do both)

In the programming portion, you may use the book *Picturing Programs*, the Racket Help Desk, your `posn-util.rkt` file, and the class blog.

Your work will be evaluated on the basis of correctness and how well it demonstrates your understanding of the design process.

Suggested time: 60 minutes.

1. **Pseudo-Roman**. The Roman numerals representing the number 203 are `CCIII`, and the number 21 is `XXI`. In this question you are asked to write a converter `pseudo-roman` that takes in a positive integer between 1 and 999 and produce a string with one "C" for each hundred in the number, then one "X" for each 10 remaining after the hundreds are removed, and one "I" for each one remaining once the tens are removed.

   Note: Actual Roman numerals are more complicated than this. You may read the Wikipedia entry, but watch the time — don't get side-tracked!

   Design and test the `pseudo-roman` function.

2. **Touch-it**. A yellow circle of radius 100 is on the screen in a fixed location.

   - You control a smaller blue circle with the mouse. (The blue circle is where the mouse points.)

- In every second that the blue circle is touching the yellow circle (at all), the player gets a point.
- Draw the big circle in a different shade of yellow when the player is going to get a point.

Extra Credit

In the place of program #2 above, you may write the **radiation circle** game.

- You move a small blue dot with your mouse.
- A red dot at a random location starts at radius 50 and shrinks its radius by 10 every second.
- When the red dot radius reaches zero, it picks a new random location and starts over
- Every second that the blue dot touches the red dot (overlapping at all), you get a point.
- Show the score in one corner of the screen.