Assignment Guidelines:

- **Submission details:**
  - Solutions to these questions must be placed in files `a1q1.rkt`, `a1q2.rkt`, `a1q3.rkt`, respectively, and must be completed in Racket.
  - All solutions must be submitted through MarkUs. Solutions will **not** be accepted through email.
  - Verify your basic test results using MarkUs to ensure that your files were submitted properly and are readable on MarkUs. **Note, however, that passing the basic tests does not guarantee that you will pass all our correctness tests.**

- **For this assignment only, you are not required to use the design recipe when writing functions. In each case, you are only required to include the function header and body. You are not required to include the purpose, contract, examples, or tests. Your grade on this assignment only is based entirely on correctness and following the stated requirements of the questions.**

- Download the interface file from the course Web page to ensure that all function names are spelled correctly, and each function has the correct number and order of parameters.

- **Restrictions:**
  - You may only use the built-in functions and special forms introduced in the lecture slides in Module 01. A list of these functions can be found on the Assignments web page: [https://www.student.cs.uwaterloo.ca/~cs115/built_in](https://www.student.cs.uwaterloo.ca/~cs115/built_in)
  - Read each question carefully to see if any additional restrictions apply.
  - Test data for correctness tests will always meet the stated assumptions for consumed values.

- The solutions you submit must be entirely your own work. Do not look up either full or partial solutions on the Internet or in printed sources.

---

Plagiarism: The following applies to all assignments in CS115.

All work in CS 115 is to be done individually. The penalty for plagiarism on assignments (first offense) is a mark of 0 on the affected question and a 5% reduction of the final grade, consistent with School of Computer Science policy. In addition, a letter detailing the offense is sent to the Associate Dean of Undergraduate Studies, meaning that subsequent offenses will carry more severe penalties, up to suspension or expulsion.

To avoid inadvertently incurring this penalty, you should discuss assignment issues with other students only in a very broad and high-level fashion. Do not take notes during such discussions, and avoid looking at anyone else’s code, on screen or on paper. If you find yourself stuck, contact the ISA or instructor for help, instead of getting the solution from someone else. Do not consult other books, library materials, Internet sources, or solutions (yours or other people’s) from other courses or other terms.

Be sure to read the Plagiarism section at: [https://www.student.cs.uwaterloo.ca/~cs115/assignments#Plagiarism](https://www.student.cs.uwaterloo.ca/~cs115/assignments#Plagiarism)
Question 1: So, I need to pass the "weighted averages" of my exams, but what does that really mean?

One of the requirements for passing CS115 is that you pass the weighted exam average for the course. Since the midterm is worth 30% of your grade, and the final worth 45%, this means that your midterm grade ($m$) and your final exam grade ($f$) must satisfy:

$$0.30 \times m + 0.45 \times f \geq 37.5$$

where $m$ and $f$ are your exam percentages converted to grades between 0 and 100. Once you know your midterm grade, you can determine the minimum grade on the final exam, which will allow you to meet this requirement by rearranging the formula:

$$f \geq \frac{37.5 - 0.30 \times m}{0.45}$$

Complete the Racket function `final-at-least` that consumes $m$, your midterm grade (any number between 0 and 100, inclusive), and produces the minimum grade by the above formula that ensures the weighted exam average is passed. For example, `(final-at-least 80) => 30`. Note that your answer will not always be an integer.

Question 2: It was so hot ...

When it is really warm, weather reports often include a humidex reading as well as the actual temperature. The formula for the humidex is below (fun fact: the formula was developed by Canadian scientists! See https://en.wikipedia.org/wiki/Humidex):

$$H = T_{\text{air}} + 0.5555 \left[ 6.11 \times e^{\frac{541.7530}{273.16 + \frac{1}{T_{\text{dew}}}}} - \frac{1}{273.15 + T_{\text{dew}}} \right]$$

where

- $T_{\text{air}}$ is the air temperature (in Celsius)
- $T_{\text{dew}}$ is the dew point (in Celsius) and $T_{\text{dew}} \leq T_{\text{air}}$
- $e$ is Euler’s number (2.718281828...). The built-in Racket constant $e$ corresponds to this special, inexact value. You may use either of Racket’s exponentiation functions (`exp` and `expt`) when writing your solution.

Complete the Racket function `humidex`, which consumes two integers, `t-air` and `t-dew`, and produces the humidex in Celsius using the above formula. The Racket calculation will give an inexact number. Inside your function, convert the calculated value to an exact number using the built-in function `inexact->exact`, and then round that result to the nearest integer (using the function `round`), so that the function produces an integer. For example,

- `(humidex 15 10) => 16`
- `(humidex 31 28) => 47.`
Question 3: *Math can definitely be fun ...*

The calculations described in this question show how to reconstruct a 10-digit phone number from parts of the number itself. **While the result of the calculations will always be the number you start with, in order to receive any marks for this question, you must do the calculations exactly as they are described below. Avoid any shortcuts in your implementation of the calculations.**

Complete the Racket function `phone`, that consumes `ph`, a 10-digit integer, performs the following calculations, and returns the resulting integer. For ease of explanation, an example is included with the steps below, but remember your function must work for any 10-digit integer `ph`.

<table>
<thead>
<tr>
<th>Step Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Determine the first 3 digits of <code>ph</code>. <em>(The first 3 digits are called the area code.)</em></td>
<td>5195550123</td>
</tr>
<tr>
<td>Multiply the result from the previous step by 40.</td>
<td>20760</td>
</tr>
<tr>
<td>Multiply the result from the previous step by 25.</td>
<td>519000</td>
</tr>
<tr>
<td>Add the middle 3 digits of <code>ph</code> to the result from the previous step. <em>(The middle 3 digits are called the exchange.)</em></td>
<td>519555</td>
</tr>
<tr>
<td>Multiply the result from the previous step by 50.</td>
<td>25977750</td>
</tr>
<tr>
<td>Add 1 to the result from the previous step.</td>
<td>25977751</td>
</tr>
<tr>
<td>Multiply the result from the previous step by 400.</td>
<td>10391100400</td>
</tr>
<tr>
<td>Add the last 4 digits of <code>ph</code> to the result from the previous step. <em>(This number will always be an even integer.)</em></td>
<td>10391100646</td>
</tr>
<tr>
<td>Divide the result from the previous step by 2.</td>
<td>5195550323</td>
</tr>
<tr>
<td>Subtract 200 from the result from the previous step. The function produces this value.</td>
<td>5195550123</td>
</tr>
</tbody>
</table>

So, `(phone 5195550123) => 5195550123`

The trickiest part of these calculations is determining the first 3, middle 3, and last 4 digits of `ph`. You can figure this out using the built-in `remainder` and `quotient` functions. Think about that first, and the rest of the calculations are pretty straightforward.

An additional example is illustrated in this video ([https://www.youtube.com/watch?v=0_EFWTG3S4Y](https://www.youtube.com/watch?v=0_EFWTG3S4Y)) (but, since it is YouTube, probably best to ignore the comments!), where the calculations are phrased as a way to "guess" someone's phone number after they have done the initial calculations.

*Note: For this assignment, you are not required to follow the design recipe (described in Module 2) or apply good programming style techniques, such as helper functions and constants. As a result, you may find some of your functions hard to follow. Throughout the remainder of the course, you will learn various techniques for improving your solutions.*