Discrete Math & Functional Programming— CSCI 054— Spring 2024
Instructor: Osborn
Homework - Week 01 (7 point(s))
Due: 10:00PM on Sunday

- This assignment must be turned in individually on Gradescope. However **for this assignment only** you may collaborate as much as you want with the other students in your small group.
- **If you ever have any questions about what is an acceptable amount of collaboration (or an acceptable use of ChatGPT or similar programs!), you must discuss with the professor in advance!!!**

1. **[7 point(s)] reading and submitting code**

   Everyone should turn in this problem on Gradescope individually, although **for this assignment only** you may work as much as you want with the other students in your group. In fact, your group is highly encouraged to work on the problem together.

   For this week's group assignment you downloaded the template file and saved it on your computer. Now open this file in your preferred editor (most likely emacs or VSCode). This is the file where you will put your solutions and then submit to Gradescope.

   The solution for the first two questions is given in the template; please take the time to look at the code and to get a general sense of how it works. Additionally we have put in comments describing each of the functions that you are expected to write. In the future you will be expected to include your own comments before the start of each function.

   (a) Write a function `numList` with the property that `numList n` is a list of integers from `n` down to `1`.

   Solution:

```
numList :: Integer -> [Integer]
numList n =
    if n <= 0
        then []
    else
        n : (numList (n-1))
```

(b) Write a function `sumFormula` with the property that `sumList n` is the sum of the integers from `n` down to `1`, but uses the mathematical formula

$$n + (n-1) + (n-2) + ... + 1 = \frac{n(n+1)}{2}$$

Solution:

```
sumFormula :: Integer -> Integer
sumFormula n = round ( (fromIntegral n) * (fromIntegral (n+1)) / 2.0 )
```

(Thought question: What happens if you just have `sumFormula n = n * (n+1) / 2`? Why do you think that is?)

(c) Write a recursive function `sumList` with the property that `sumList n` is the sum of the integers from `n` down to `1`. `sumList` should have the type signature `Integer -> Integer`. If `n` is less than 1 the function should return 0.

(d) Write a function `sumCheck` that takes an Integer `n` and returns `True` if `sumFormula n` and `sumList n` return the same value. `sumCheck` should have the type signature `Integer -> Bool`

(e) Write a function `min3` that takes a triple of numbers (a tuple with 3 elements, **not** a list!) and returns the smallest of the three. For example, `min3 (1,5,-2)` should return `-2`. The type signature should be `(Ord a) => (a, a, a) -> a`. (Why is this the type signature?) You may **not** use the built-in `min` function.

Look for the assignment `week01-ps-coding` on Gradescope and upload the Haskell file that contains your solutions. There is an autograder on Gradescope which will run a single test case for each function. This will ensure that all your types are correct. Note that when we actually grade your assignment we will run many more tests, so you should make sure to test your code thoroughly!