PLC: Workout 1 [80 points]

Due date: Friday, January 28th by midnight

About this homework

You will get started with Haskell, and solve some basic programming problems in Haskell.

Starter code, submission via Github Classrooms

We are going to use Github Classrooms for this assignment (and hopefully the others, too). So you will need a git client for your computer, to download repositories for homeworks and push your solutions back. If you do not have a git client, you might like to try `https://git-cola.github.io`, which is free, allows private repos, runs on all platforms, and provides a nice interface.

You will need a github login. Please create one if needed (it is free and easy). Then to start the homework, visit this url:

`https://classroom.github.com/a/pKvfxckJ`

You will be asked to select your Hawkid from a long list. This will associate your github login with your Hawkid on Github Classrooms. Github will then create a new repo for you with the starter code for workout1. You can clone this to your computer using your git client in the usual way. (We will review this a little at the start of class January 25th.)

You will submit your assignment by pushing commits to your repo. You can commit as much as you want up to the deadline. We have some tests set up to run on each commit. You can see the results under the "Actions" tab on the github web page for your workout's repo.

How to get help

You can post questions in the Discussions section of ICON.

You are also welcome to come to our office hours (Zoom only). See the "Office hours" page under "Pages" on ICON for the office hours for the course staff:

`https://uiowa.instructure.com/courses/179776/pages/office-hours`

1 Reading

Read Chapters 1, 2, and 3 of the required book, *Programming in Haskell*, by Graham Hutton.

2 Getting started with Haskell [15 points]

For the first part of this course, you will be programming in Haskell. While you are welcome to use Haskell from the CS lab Windows machines, I encourage you to install these on your own computer if possible. If you need to use the CS lab Windows machines, here is a page describing how to access them remotely:

`https://clas.uiowa.edu/linux/services/vmwareview`

Installing Haskell

Haskell is freely available for Windows, Linux, and Mac. For Windows, we have created a single installer that provides everything you will need now and later in the class, here:

`http://homepage.divms.uiowa.edu/~astump/agda/Agda2.6.0.1.v1.msi`

If you use the installer, you should probably restart your computer after you apply it, or Windows may have trouble finding Haskell.

You can also install Haskell following the directions at `www.haskell.org`. This works on all three major platforms. I am using Emacs in class for editing Haskell files, and for using Agda later in the course you will need Emacs. But for now, you can edit `.hs` files using any plain-text editor, like Notepad++ on Windows.

If you have problems getting Haskell working on your computer, please see us in office hours, and we will help you.

2.1 Testing Haskell [15 points]

To make sure that Haskell is working correctly, you first have to open up a terminal (Mac, Linux) or command shell (Windows). On Windows, you can start a command shell by searching for `cmd` and then selecting "Command Prompt". Now navigate to your copy of the workout1 repository (you can use `dir` and `cd` on Windows, or `ls` and `cd` on Mac/Linux), and enter `ghci`. This should start the Haskell interpreter (if it does not, it likely means your path is not set correctly to include the directory where Haskell is installed on your computer). Then type `:l Basic`, to ask the Haskell interpreter to load `Basic.hs`. You should see something like this (likely with a more recent version number for Haskell):

```
antioch:~/plc-priv/hw1$ ghci
GHCi, version 8.6.5: http://www.haskell.org/ghc/  :? for help
Prelude> :l Basic
[1 of 1] Compiling Basic            ( Basic.hs, interpreted )
Ok, one module loaded.
*Basic>
```

Now type `hello` and hit enter. You should see something like this:

```
*Basic> hello
"Welcome to Haskell!"
```

Please take a screenshot of this interaction called `screenshot.png`, and add it to your submission folder. This is just to demonstrate that you were able to start up ghci. There is a test for existence of this file under Actions on the github page for your repo (so if you push a commit and you have that file there, you will see that test pass).

2.2 Haskell exercises [65 points]

Complete the 13 exercises in `Basic.hs`. Each is worth 5 points. There are tests in `PublicTests.hs` you can use to help make sure your code is correct. You can use `ghci` to try out your code, and run the tests. Just do `:l PublicTests`, and enter `main` to run the tests. You will see some output for the problems you solved so far, and then some error message for the problems you did not solve yet, like this

```
*** Exception: Prelude.undefined
CallStack (from HasCallStack):
  error, called at libraries/base/GHC/Err.hs:78:14 in base:GHC.Err
  undefined, called at ./Basic.hs:16:14 in main:Basic
```

Once you solve all the problems, that error message will go away.

There is a test under Actions to make sure your code compiles, and one to see if it passes the public tests (so again, if you push commits to github, you can check under Actions to see if the tests are passing).

Grading

Your grade will be determined by whether the required `screenshot.png` file is submitted, and how many public tests your code passes. We might also use some private tests. Code that does not type-check will not receive partial credit, so make sure that your code at least type-checks with Haskell (in other words, Haskell does not print type errors when you load the file). The `compile` test we have under Actions will show you whether your code is compiling or not, once you push to github.