PLC: Homework 4 [100 points]

Due date: Wednesday, April 3rd
3 extra-credit points if you turn in by Tuesday, April 2nd

About This Homework

For this homework, you will start learning Agda by writing some proofs. You will first install Agda on your computer (optional if you instead use the CS Windows lab computers), and the Iowa Agda Library.

How to Turn In Your Solution

You should create a `hw4` subdirectory in your github.uiowa.edu repo. You will copy files from subdirectories of the `hw4` directory in the course repo.

Partners Allowed

You may work by yourself or with one partner (no more). See the instructions from `hw1` for details on how to submit your assignment if you work with a partner.

How To Get Help

You can post questions in the `hw4` section on Piazza.

You are also welcome to come to our office hours. See the course's Google Calendar, linked from the Resources tab of the Resources page on Piazza, for the locations and times for office hours.

1 Reading

Read Chapters 2, 3, and 4 of Verified Functional Programming in Agda, available for free (on campus or VPN) here:

`https://dl.acm.org/citation.cfm?id=2841316`

2 Installing Agda

Agda is installed on the CS Windows computers. You will probably want to install it also on your own computer. For Windows, the easiest thing is to use our installer (which we have updated now and it works):

`http://homepage.cs.uiowa.edu/~astump/agda/AgdaBundle_2.5.4.2.v4.msi`

Otherwise, try following the directions on the Agda wiki, here:

`http://wiki.portal.chalmers.se/agda/pmwiki.php`

Essentially you first do `cabal install Agda` and then `agda-mode setup` (the latter probably requires that you add `~/.cabal/bin` to your path). If you install Agda this way, you should get Agda version 2.5.4.2 (any version 2.5.4.x is ok).

3 Installing the IAL

You clone the repo here from github:

`https://github.com/cedille/ial`

4 Configuring and testing Agda and the IAL [20 points]

Finally, you need to tell Agda how to find the Iowa Agda Library. If you are using a CS Windows machine, then open the file `h:/.emacs`. Otherwise, open `~/.emacs`. Add the following text, where instead of the word `PATH`, you should have the path to your copy of the IAL (wherever you put it):

```
(custom-set-variables
 '(agda2-program-args (quote ("--include-path=PATH"))))
```

That should be a single forward tick mark on the second line of that code (might render incorrectly in this PDF). On Windows, I found I could put backslashes if I escaped them (double backslash), like this (where `Myself` is, of course, your actual Windows username):

`C:\\Users\\Myself\\Documents\\ial`

To prove that all this is working for you, open `bool.agda` in the IAL and type Control-c Control-l to load the file with Agda. If this succeeds you should get syntax highlighting for the file. Now

take a screenshot called `ial-screenshot.YYY`, capturing your Emacs window with `bool.agda` highlighted. (I found that for some reason, Agda often says "Another command is currently in progress" when I do this, and I must first type Control-c Control-x Control-r to restart Agda, and then do Control-c Control-l.)

5 Boolean theorems [30 points]

In `bools.agda` in the `hw4` directory, you will find five lemmas to prove. When you load the file with Control-c Control-l, you will see holes on the right-hand sides of the definitions of those lemmas. Remove those holes (Control-k with your cursor right before the hole will cut it out), and fill the definitions in with proofs. [6 points each]

6 Simple equational theorems [25 points]

In `simple.agda` you will find five problems similar to ones we did in class March 14th (see the lecture notes). They are worth 5 points each.

7 Theorems about list operations [25 points]

In `list-todo.agda` you will find definitions of two functions `space-every-other` and `swap-pairs`. After these definitions there are five holes to fill in: four proofs and one condition (in the statement of `nuke-all`) that has to be selected so that the last theorem is provable. The condition should be chosen so that `test` at the very bottom of the file type-checks without yellow highlighting. Each hole is worth five points.