PLC: Homework 4 [100 points]

Due date: Wednesday, April 4th, 9pm
3 extra-credit points if you turn in by Tuesday, April 3rd, 9pm

About This Homework

For this homework, you will start learning Agda by writing some programs and proofs about them. You will first install Agda on your computer (optional if you instead use the CS Windows lab computers), and the Iowa Agda Library. Note that depending on which version of Agda you install, you will need to use either the 1.3 release or 1.4 release of the Iowa Agda Library.

How to Turn In Your Solution

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

As for previous homeworks, you can check that you have submitted correctly by going to the URL for your subversion repository. Remember to use exactly the file names we are requesting (so do not change the names of these files).

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

`https://dl-acm-org.proxy.lib.uiowa.edu/citation.cfm?id=2841316&CFID=852046702&CFTOKEN=22704153`

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/Agda2.5.2.v2.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 will get Agda version 2.5.3.

3 Installing the proper version of the IAL

Next, you have to check out the Iowa Agda Library (IAL). If you installed Agda version 2.5.2, you need to use this version of the IAL:

`https://svn.divms.uiowa.edu/repos/clc/projects/agda/ial-releases/1.3`

If you installed version 2.5.3, you need to use this version of the IAL:

`https://svn.divms.uiowa.edu/repos/clc/projects/agda/ial-releases/1.4`

In either case, the username and password are both "guest" (no quotes).

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"))))
```

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 Simple theorems [40 points]

In `simple.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, using the assumptions ("postulates") at the start of the file. [8 points each]

6 List code and proofs [40 points]

See `list-todo.agda` for these problems. One tip (which we will see in class, too): to see how to type a Unicode symbol in Agda mode, you can type "Alt-x" and then "`describe-char`", with your cursor on that character. Alternatively, I believe the current Agda mode lets you type Control-c Control-k for this.

1. Fill in the definitions of `take` and `intersperse` so that they behave like the Haskell versions of these functions (see the documentation for the `Data.List` module on hackage). [10 points each]
2. Fill in the definition of `length-take`, to prove that the length of the list you get by calling `take n xs` is always less than or equal to `n`n. For this, I found I needed a lemma from `nat-thms.agda` about less-than-or-equal-to (type `\le` in emacs to type that $\leq$ symbol). [10 points]
3. Now you will prove a similar theorem about the length of the list returned by intersperse. First figure out what the type of `length-intersperse` should be and fill that in [5 points], then prove it [5 points].