CS 3110 Fall 2014 **Due at 11:59 PM, 12/11/14, no late passes**
Problem Set 7
Version 1 (last modified November 21, 2014)

Optional

This problem set is optional. You do not have to submit it. In return, the course staff will supply minimal support. You are essentially on your own. Have fun!

Objectives

- Learn a new functional programming language.
- Use mechanized logic in a proof assistant.

What we supply

We provide an archive `ps7.zip` that you should download from CMS. It contains several `.v` files, which are Coq source files.

What to turn in

You will modify the provided `.v` files in creating your solutions. Submit the modified files.

Teams and source control

This problem set must be done individually. You may discuss solutions in general terms with other students, but you may not work on code with them or share code with them. You do not need to turn in a git log.

Getting help

Few (none?) of the TAs or consultants have ever seen Coq. Please avoid using consulting or office hours on Coq if there are other students waiting to talk about required course material. The instructor will attempt to answer Piazza posts about Coq, but they will receive lower priority than required course material.

How to install Coq

You can download and use CoqIDE natively on many platforms. Here's how to install it in the CS 3110 VM:

```
sudo apt-get install libgtk2.0-dev
opam install coq coqide
```

Expect the second command to take a long time. You can run CoqIDE by typing `coqide` at the command line.

What to do

[code] The `.v` files we supply are chapters from a textbook by Benjamin Pierce et al. titled *Software Foundations*. We have given you these chapters:

1. `Preface.v`
2. `Basics.v`
3. `Induction.v`
4. `Lists.v`
5. `Poly.v`
6. `MoreCoq.v`
7. `Logic.v`
8. `Prop.v`
9. `MoreLogic.v`
10. `ProofObjects.v`

These chapters constitute about one month's worth of material in a senior-level class. They start with functional programming in Coq, and move on to logic. By the time you get to `Logic.v`, you will recognize some of the material we covered in class.

You should open each chapter file in CoqIDE,[1] read the file, and solve the exercises that you encounter. To solve an exercise, simply modify the file to provide your solution. Note that `Preface.v` does not contain any exercises.

Each exercise has a number of stars, from 1 to 5. Some exercises are tagged as "advanced" or "optional." And some exercises expect written (English) answers instead of Coq code. For this problem set, we will only be concerned with the non-advanced, non-optional, non-written exercises. Let's call this set the *available exercises*.

You are free to skip some exercises and solve others, though you should be aware that the chapters build on one another. We strongly recommend that you do all the 1-star exercises as you proceed through the chapters—they are designed to be quick checks that you understand

[1] You may instead use the ProofGeneral interface for Emacs.

what you are reading. We also recommend that you skip all the 4 and 5 star exercises at first—they are likely to be too time-consuming to be worthwhile.

You may not use the `firstorder` or `crush` tactics that we saw in class. They would defeat the purpose of you learning some Coq, because they would automate too much for you.

Grading

If a `.v` file does not compile, you will receive no credit for it. If you use the `admit` keyword in a definition, or the `Admitted` keyword in a proof, your solution to that exercise and any exercises that depend on it will receive no points. (Those keywords tell Coq to accept a definition or proof even though the definition or proof is incorrect.)

Every exercise you solve will get you a number of points equal to the number of stars of the exercise. Solving all the available exercises is worth a bonus of 2% to your final grade. Please note that this bonus is so small—about the size of a single problem on the final exam—that pursuing it makes little sense unless you just find this material interesting. Your time is probably far better spent on studying for the final.

We are aware of a couple online sources that have unfortunately posted solutions to an old version of this textbook. Copying those solutions is a violation of academic integrity. We will include those solutions as input to our plagiarism detection software. So it's not worth your effort copying them.

Comments

[written,ungraded] We invite you to submit any comments you have about the problem set or about your solutions in a plain text file named `ps7comments.txt`. We're especially interested in whether you think this material would be suitable for inclusion in (an honors version of) 3110.