Homework 9

Due Thursday, 11/21/2013

Please turn in your homework solutions online at `http://www.dci.pomona.edu/tools-bin/cs131upload.php` before midnight on the due date.

1. (20 points) **Modularity of Concrete Data Types**

   In lecture 11 we discussed an environment-based interpreter for PCF. It is available on-line from the course web page by following the link to "Programs from lecture". In this problem, I would like you to modify that program in order to determine how easy it is to add new features.

   (a) Please add a new function `prettyprint` that takes a PCF term and returns a string containing a nice readable version of the term.

   Note: The data type `Term` is given in the PCF parser, available from the same programs page. Feel free to copy and paste the definition into `PCFEnvinterp.hs` and drop the `import` line. Alternatively, turn in the `ParsePCF.hs` file as well (enhanced as needed in part b), but don't forget to add `AST_SUM` (see part b) to the export list of that module.

   The string should contain parentheses to indicate precedence. For example, the term

   ```
   AST_IF(AST_BOOL True,AST_NUM 5, AST_APP(AST_SUCC, AST_NUM 2))
   ```

   should be pretty-printed as

   ```
   if true then 5 else (succ 2)
   ```

   whereas

   ```
   AST_APP(AST_IF(AST_BOOL true,AST_PRED, AST_SUCC), AST_NUM 2)
   ```

   should be pretty-printed as

   ```
   (if true then pred else succ) 2
   ```

   (b) Please add a new term to PCF representing sums of integers. I.e., add a new clause to the definition of term of the form `AST_SUM (Term, Term)` so `AST_SUM(a,b)` represents `a + b`.

   Modify the previous program that contains both the `newinterp` function to evaluate terms and `prettyprint` to accommodate the new term.

   (c) I am not asking you to modify the PCF parser code that is available on-line. However, please describe at a high level what would have to be done to modify that code.

   (d) Discuss briefly the different impacts of adding a new function (like `prettyprint`) versus adding a new term (like `AST_SUM` to the data type in the previous two parts of this problem.

   Note: If you wish, you need only turn in the code from the part b of this question rather than separate code for each of parts a and b. Of course you should also include your answers for parts c and d.

2. (15 points) **Subtyping and Visibility**

   Please do problem 12.6 from Mitchell, page 375.

3. (10 points) **Subtyping and Specifications**

   Please do problem 12.9 from Mitchell, page 377.

4. (25 points) **"Growing a Language" Reading**

   Please read the short paper entitled "Growing a Language" from the auxiliary reading page. The author, Guy Steele, is one of the designers of Java, and this was his keynote talk at the major American conference on object-oriented languages (OOPSLA) in 1998. You may find it even more interesting to watch his talk on-line (see the link on the auxiliary reading page). See if you can figure out what he is doing before he explains it!

   This paper includes a discussion of why some items were originally left out of Java when they designed the language, but likely should have been included. It also gives some insight into programming language design in general. The following questions will guide you through the paper:

   (a) What is wrong with designing/using a small language like Lisp?

   (b) What is wrong with designing/using a huge language (C++)?

   (c) What was Steele's goal in designing Java?

   (d) Why does Steele feel that overloaded operators are important. Contrast this with his conviction (expressed elsewhere) that overloaded methods might have been left out without much harm.

   (e) Discuss why generics might have been left out originally, but why he now feels they are important.

   Write a separate paragraph discussing each of these items. A few sentences on each part is sufficient.

   We will spend some time discussing these points after the homeworks are turned in.

5. (15 points) **Smalltalk Reading**

   Read "Design Principles Behind Smalltalk" by Dan Ingalls (available on the "links" page).

   Think about the Principles set forth. Which are new? Which have we seen before? What principles are remniscent of Lisp? What is the scope of the language (ie, what is included in its definition)? A few sentences on each item is sufficient— clarity is more important than length of answer.