Homework 3: Dataflow Analysis and Project Work

In this homework you will learn more about dataflow analysis and also work on your semester project.
The project portion of this homework is intended to be done in your project groups. Be sure to follow the process described in the course's grading policy if you work in groups.

1. (20 points) [Concepts] [Calculate] [Semantics] Do exercise 1.6. In your proof use a calculational style (as in the handouts we provide in class [1, Section 4.2] [2, Chapter 4], see also Gries's article in *CACM* [3]), in which you justify each step. If you want to use LaTeX for solving such problems, you might like to use the macros we provide for calculational proofs and program analysis. These can be obtained from `http://refine.eecs.ucf.edu/gf/project/tex-include/scmsvn/` by browsing into the folders: trunk, then texmf, then tex, then latex, then misc; then the files you want are `calculation.tex` and `program-analysis.tex`.

2. (10 points) [Concepts] [Semantics] According to our textbook [4], when using Chaotic iteration to find a solution for the Available Expressions analysis (in Section 2.1.1), one should start with a tuple in which each element is $\mathbf{AExp}_\star$. This problem is about creating an example to show why that is necessary.

   Create an example program in the WHILE language for which the Chaotic iteration of the function that represents that program's Available Expressions analysis does not get the right result, if the iteration starts with $\vec{\emptyset}$. (Hint: the next part of this problem will be easiest if your example is very small.)

3. (10 points) [Concepts] [Calculate] [Semantics] Show how the Chaotic iteration for your example in the previous problem gives the right result for the Available Expressions analysis, if you start the iteration with a tuple in which each element is $\mathbf{AExp}_\star$. This should be expressed in the form of a calculation.

4. (10 points) [Concepts] [Calculate] [Semantics] Give an example program in the WHILE language in which the chaotic iteration for the Very Busy Expressions analysis (of section 2.1.3) does not give the right result, if the iteration starts with $\vec{\emptyset}$. To demonstrate your answer is correct, show what happens using chaotic iteration for that example with initial value $\vec{\emptyset}$.

5. (10 points) [Concepts] [Semantics] What property of an analysis, in general, determines what starting value is appropriate for iterations used to find a solution?

6. (20 points) [Concepts]

   Do exercise 2.3 in our textbook [4].

7. (150 points) [Concepts] [BuildTools] Write and test one static analysis for your semester project. Turn in enough so that we can understand what you have done. Appropriate overview and/or comments will be helpful. Hand in your source files (but no generated files!).

   We recommend that you use JastAdd (see `http://jastadd.org`). For working with JastAdd, you should read the on-line documentation, especially the reference manual. Also, it's useful to start with an existing JastAdd sample project (see `http://jastadd.org/projects`), such as PicoJava, and modify it to suit your language.

   You can use our WHILE language as a sample project also. The Unix command for anonymous checkout is:

   ```
   svn checkout http://refine.eecs.ucf.edu/svn/proganalysis/WHILE/trunk WHILE
   ```

   and the URL `http://refine.eecs.ucf.edu/svn/proganalysis` can be used with tools like Eclipse, but check out the WHILE/trunk if you are working with Eclipse. We also make available a snapshot of the project in `http://www.eecs.ucf.edu/~leavens/WHILE.tgz`.

References

[1] Ralph-Johan Back and Joakim von Wright. *Refinement Calculus: A Systematic Introduction*. Graduate Texts in Computer Science. Springer-Verlag, Berlin, 1998.

[2] Edsger W. Dijkstra and Carel S. Scholten. *Predicate Calculus and program semantics*. Springer-Verlag, NY, 1990.

[3] David Gries. Teaching calculation and discrimination: A more effective curriculum. *Communications of the ACM*, 34(3):44–55, March 1991.

[4] Flemming Nielson, Hanne Riis Nielson, and Chris Hankin. *Principles of Program Analysis*. Springer-Verlag, second printing edition, 2005.