Homework 1: Introduction to Program Analysis

See the syllabus for the due dates.
In this homework you will get an overview of program analysis and make some initial plans for your semester project. If you wish, you can work in groups, and I recommend that for the semester project. However, but be sure to follow the process described in the course's grading policy if you work in groups.

Read chapter 1 of our textbook: *Principles of Program Analysis* [1]. There are also some online resources about program analysis listed on the course resources web page,
`http://www.cs.ucf.edu/~leavens/COP5021/resources.shtml`. You can also look for new books or online resources that are interesting. (Hint: to find new resources, look in Chapter 1 of the textbook for terms to search for. You may find more material under the specific approaches, such as data flow analysis or abstract interpretation, than under the general term "program analysis.")

1. [Concepts] Give a very brief answer (a sentence or two, at most a paragraph) to answer the following.
   (a) (10 points) What are (i) the main ideas and (ii) the main goals of program analysis?

   (b) (10 points) Aside from optimizing compilers for programming languages, briefly describe (a) one other area of computer science in which the ideas and goals of program analysis might be usefully applied, and (b) how the ideas might apply in that area.

2. [BuildTools] Write a short proposal (about 2 pages in length) for a semester project that involves writing a static analysis tool to answer some question about software. This proposal should describe:
   (a) (4 points) The overall problem you wish to solve, and why it is important or interesting.
   (b) (7 points) The static analysis question[1] that your tool will answer. Also give one or two brief examples of programs that illustrate this question and explain why this question can be answered statically (without executing the code).
   (c) (3 points) The language your tool will analyze. This can be a programming language, a specification language, or some other kind of language (such as a database query language) that is similar to a programming language.
   (d) (6 points) Any related work on this problem that you can find, and why the related work does not answer the analysis question you are asking adequately.

   It is best if you restrict your project to an intraprocedural analysis; that is, it is best if your analysis does not need to use analysis results from outside of a given procedure (or method) when doing analysis on a single procedure (or method) body.

   Our advice is to avoid choosing a language that is very complex or that has a syntax that is difficult to parse (so don't do C++). Also, it will be best if you choose a language that is a first-order language (without procedure or function values), like the WHILE language of the textbook, because handling higher-order languages (e.g., Scheme or Haskell) is a topic that we will likely not cover in detail this semester (but see chapter 3 of the textbook).

   In any case, the language your project analyzes *cannot* be the WHILE language of the textbook [1]. However, it may be a very restricted (small) subset of a real language. To describe a language subset, either give a grammar for that subset or list the features that you do not allow in the subset (e.g., Java without exception handling and parallelism).

   It is recommended that you meet with the instructor to discuss this proposal, and that you do so as a group if you are working in a group. However, if you cannot meet in person, email or a phone call can be used.

[1] Some examples of analysis questions appear in the textbook's description of the analyses described in chapter 2, near the beginnings of sections 2.1.1, 2.1.2, 2.1.3, and 2.1.4.

References

[1] Flemming Nielson, Hanne Riis Nielson, and Chris Hankin. *Principles of Program Analysis*. Springer-Verlag, 1999.