Adamson, Benjamin
Monday February 6
CS480 – Winter 2012

Milestone Report 2

What do I think the purpose of this milestone is?

To have us discover what lexical analyzers are responsible for doing. Spending time figuring out what belongs in the parser and what belongs in the lexical analyzer. The milestone also forced us to think about how to test our lexical analyzer. I also believe that implementation presents it's own set of unique challenges, so part of the purpose of this milestone is to deal with the those challenges.

How did our team go about solving the problem?

We started our developing a data-structure for each token that we never ended up using. After that we developed finite-state automata for each type of input, initially on paper. Once we had our paper implementation of our finite state automata finished we implemented them within separate static classes. From there we used each of the automata's to develop a scanner algorithm. During the scanner's implementation mass unit-tests were written to ensure the lexical analyzer works correctly.

Testing requirements (how did you go about testing for correctness)?

We ended up writing a sum-total of 41 unit tests that test various inputs to the lexical analyzer. We tested expected inputs against expected outputs, verifying the lexical analyzer works on all of our test inputs. The most rigorous testing was done on string inputs, as they are the most complex to analyze. The most difficult test-cases to get correct were strings with escaped quotation characters. Among our unit tests we also tested the functionality of the finite state automata's. The unit tests run automatically on our windows build, using visual studio's built in testing suite.

Retrospective

I learned which parts of the compilation process belong in the lexical analyzer, by our design. I also learned that the symbol table should be created within the parser, as according to the book. However, this will be verified when we go implement the parsing stage of our compiler. Also the amount of errors that can crop up during lexical analysis was surprising. I've never compiled a C# program in a Linux environment, so learning about the process of getting the mono compiler setup, and the program to handle input files depending on what OS is running was an enlightening task.