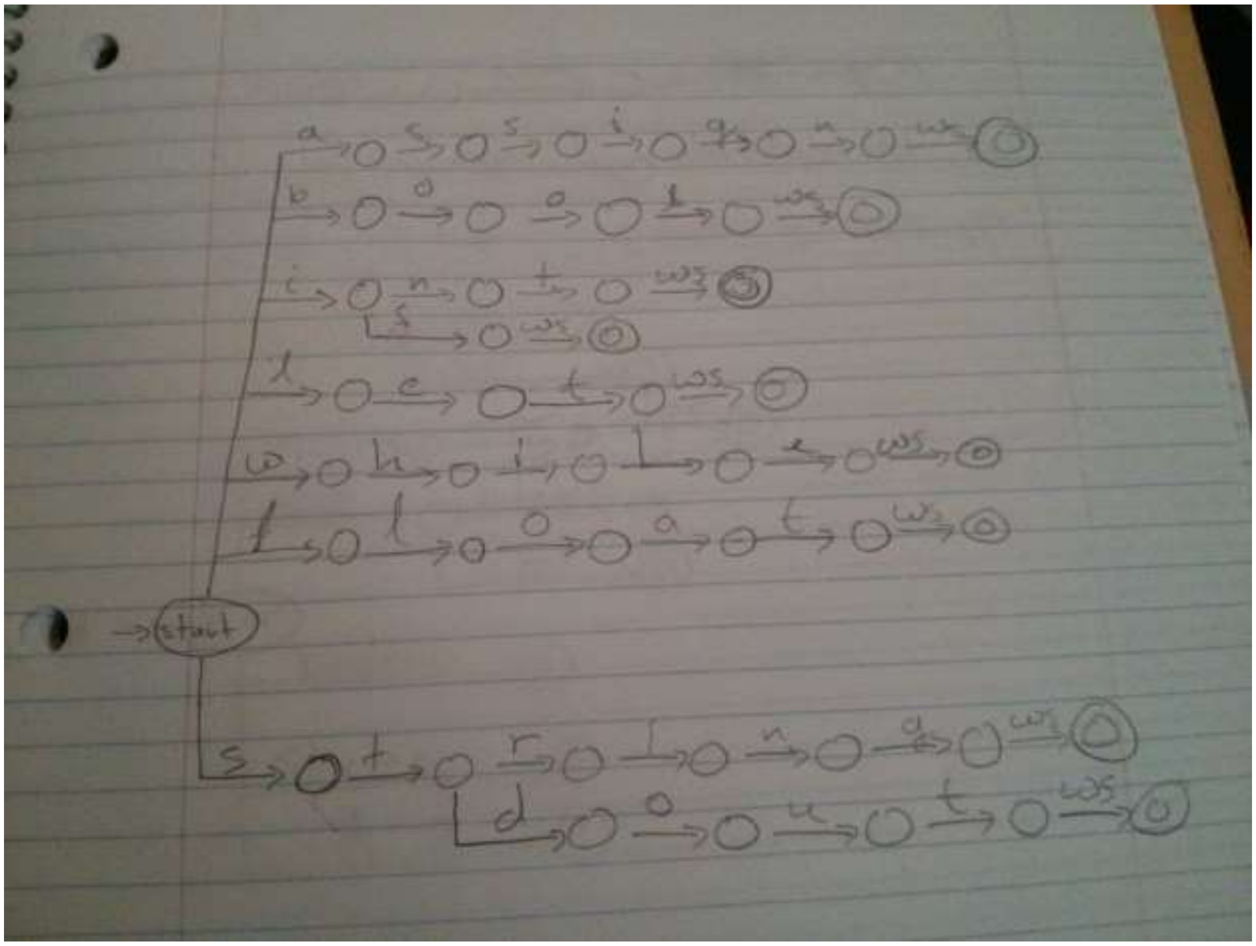
Keenon Ono
2/4/14
Milestone 2

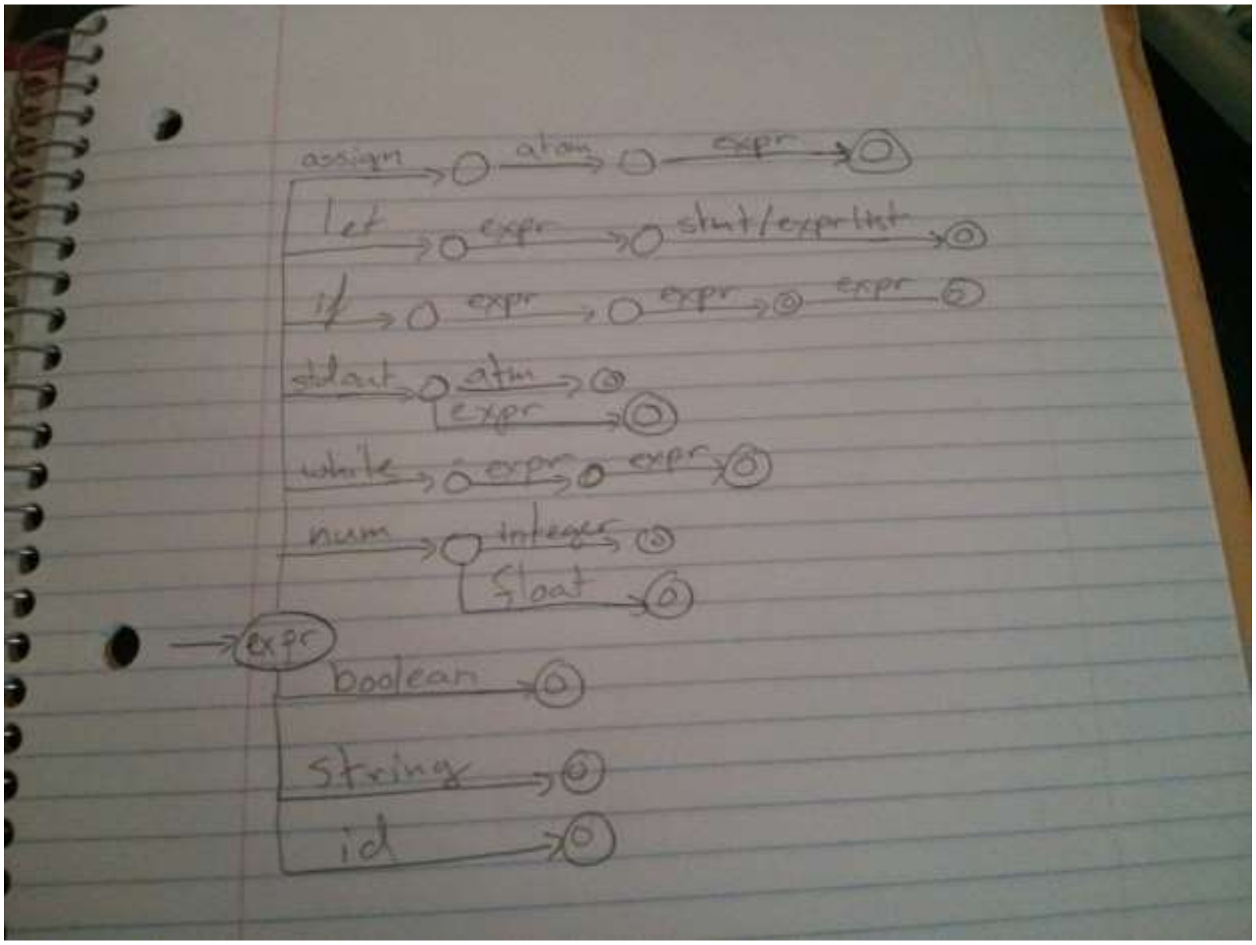
1. I am programming in Java, and will be using the inheritance hierarchy method:

```
expression ( has attrib tokenId )
      constant ( abstract )
            boolean ( not and or iff )
            number ( abstract ) ( + - * /  ^ % = < )
                  int
                  float
            string ( + = < )
      statement ( abstract )
            stdout
            if
            while
            let
            assign
```

2.



3.



4. **Purpose**
   The purpose of this milestone was to familiarize ourselves with the concept of lexical analysis. We were tasked with creating an analyzer for the IBTL and then to create a tokenizer to create tokens out of the input strings and assign a base level type. The output from the program and corresponding input file are the tokens and believed types of those tokens.

5. **Processing**
   The input stream is broken up in lines, then by whitespace. I am using whitespace delimitation in order to easily create characters to be tokenized. The other characters that I am delimiting on are quotes and left and right square brackets. Instead of tokenizing strings as words, I have tokenized the whole string so that the string will be read as its own token.

6. **Testing**
   For testing, I created an input file, stutest.in, that goes through many test cases and puts all output in the stutest.out file. Each line in the output file is the token of some lexeme and the type that was calculated using my lexical analyzer. I felt that I was very thorough with my test cases and looked for edge cases and was able to identify things in the scope of this assignment correctly.

7. **Retrospective**
   Looking back at this assignment, I probably should have started it earlier. I did have it working using regular expressions, but had some issues with a few cases and odd behavior when using the analyzer that I wrote. I think in the future I would be able to do this much quicker, but I did have some odd behavior where comparisons weren’t returning the values I expected so I had to include some extra checking. Hopefully in a real world situation, I would be allowed to use regular expressions, especially since many in largely used languages have been extensively tested.