IN2009 Tutorial 4

Download and unzip the IntelliJ project Tutorial4.zip. As usual, you will need to edit the module dependencies and CLASSPATH to match your local setup. The `.sbnf` files referred to in the following questions are in the project's `data` folder.

1. Look at the grammar `colours.sbnf`. You will find the corresponding token definitions in class lang.colours.ColoursTokens. This grammar is LL(1).
    a. Determine which non-terminals are nullable (if any).
    b. Calculate FIRST and FOLLOW sets for each non-terminal. Use these to calculate selection sets for each of the six productions (rules) in the grammar.
    c. Implement a parser as class lang.colours.ColoursParser (a template is provided).
2. Look at the grammar `loopy.sbnf`. Token definitions are in lang.loopy.LoopyTokens. This is very similar to the previous grammar. Can you see how to implement it?

3. Look at the grammar `array.sbnf`. Token definitions are in lang.array.ArrayTokens. This grammar is *not* LL(1) (you should be able see some obvious choice conflicts by a quick inspection of the rules). Transform the grammar by *factoring* the conflicting rules. Check that your new grammar is LL(1).

4. Look at grammar `turtle.sbnf`. Token definitions are in lang.turtle.TurtleTokens. This grammar is *not* LL(1), but it may not be obvious why at first sight.

    a. Calculate the selection sets for the two TCmdList rules.

    b. The grammar is actually ambiguous. Demonstrate this by writing down an example sentence in the language together with two different parse trees.

    c. Suggest two ways to remove the ambiguity:

        i. Change the *language,* in a way that you think likely to be faithful to the language designer's original intentions.

        ii. Transform the grammar, so that it generates the same language but is no longer ambiguous.