**Transformed grammar into LL**

* Original grammar: see “LL1.grm” file
* Grammar with semantic symbols: see “LL1\_sdt.grm” file

**FIRST and FOLLOW sets**

* First set: see “LL1.grm.first” file
* Follow set: see “LL1.grm.follow” file

**Design**:

* Grammar.java: read “.grm” and store terminal, nun-terminal, and production information into data structures
* Production.java: contains one left-hand-side SyntaxSymbol (non-terminal) and a set of right-hand-side Symbol.
* Symbol.java: parent class for SyntaxSymbol and SemanticSymbol
* SyntaxSymbol.java: symbols used in productions in original LL1 grammar
* SemanticSymbol.java: symbols used to generate AST
* ASTNode.java: node in AST with leaf and interior node two types. Only leaf nodes store token information.
* Parse.java: read “.first” and “.follow” and store first and follow sets into a HashMap. Implementation of parsing algorithm.
* SemanticAction.java: create ASTNode and merge nodes in semantic stack based on given semantic symbol.

**Use of tools**

* grammartool.jar: remove repetition and left recursion in grammar and generate first set.
* Ucalgary tool (https://smlweb.cpsc.ucalgary.ca): generate follow set and verify parsing table.
* Gephi: visualize produced AST and verify its correctness.