Assignment 3

In this assignment, we are to experiment with Syntax Directed Definition (SDD) using Antlr in order to implement a fully functional compiler for a given language.

Consider the following grammar. It has several incomplete parts.

|  |
| --- |
| prog : block;  expr returns [int val]  : ...  ;  repeatStmt  : 'repeat' expr '{' block '}'  ;  block  : stmt+  ;  stmt  : printStmt  | assignStmt  | repeatStmt  ;  printStmt  : 'print' '(' exprList ')'  ;  exprList  : (expr ',')\* expr  assignStmt  : 'let' ID "=" expr  ; |

Here is a sample program:

|  |
| --- |
| let x = 10 let y = 20 print (x, x+1) let x = 0 repeat 5 {  print (y, x + 100)  let x = x + 1 } |

The sample output should be:

|  |
| --- |
| 10 11 20 100 20 101 20 102 20 103 20 104 |

The compiler for the proposed language.

1. Implement a compiler for the given language using SDD in Antlr.
2. Your grammar must be named Simple.g4.
3. You must implement a separate compiler class SimpleCompiler.java

The compiler must take a source file (such as the given sample), and print valid Java bytecode in the Jasmin assembler format to the console. The resulting bytecode is for an executable Java class whose name is determined by the filename of the source file.

|  |
| --- |
| # compile the src file to assembly file sample.j $ java Compiler1 sample.src > sample.j  # assemble the .j file to the executable Java class sample.class $ java -jar jasmin.jar sample.j  # run the assembled Java class file $ java sample 10 11 500 100 101 102 103 104 |

Submission

1. SimpleCompiler.java
2. Simple.g4

Marking scheme:

1. Correctness of grammar [100/3]
2. Correctness of compiler [100/3]
3. Correctness of generated assembly file [100/3]