CS335A Project Milestone 3

Aditya Ranjan 190068
Shorya Kumar 190818
Akansh Agrawal 180050

Tools Used: The tools utilized include Flex , Bison and Graphviz.

1) Flex (Fast Lexical Analyzer Generator) : It is a tool used for generating scanners: programs which recognize the lexical patterns in the text. In our codebase we used it to create tokens for the Java language which are utilised by the Bison. To create an AST Node for its utilisation in AST generation we made a FixedNode function with data type Node* . This basically is used in the action rule inside the pattern matching of each token to get the node generated in the required fashion.

2) Bison : It is a general purpose parser generator based on the LALR(1) grammar which forms the bottom up parser. In this we again utilised the Node* data type to create the Nodes. The action for CompilationUnit includes the building of tree using buildTree function for generating AST and the nodes included in the tree are generated along with their children in the actions of the subsequent production rules.

3) Graphviz : It is a graph visualization tool to visualize the AST. It includes two components: the language DOT (for describing the AST) and a tool called dot.

Building
In the terminal do the following:
$ cd src
$ make

Execution
$ cd /path/to/root
$ ./ASTgenerator −−input $< input\_file >$ −−output $< output\_ps\_file >$

For example in the above code if the $< input\_file >$ is array.java inside the tests folder and the $< output\_ps\_file >$ is graph.ps then the last command becomes
$ ./ASTgenerator −−input tests/array.java −−output graph.ps

Note: The src folder contains the files comprising lexer, parser , 3AC, symbol_table the Makefile. The make command above runs the Makefile and then we run the AST generator after coming out of src folder. A dump of the CSV file gets genrated for the symbol table and a text file for the 3AC generation.

Our implementation supports the required command line parameters input , output, help and verbose.
−−help displays the help screen which shows the usage of the AST generator program and about the various options supported.
−−verbose provides the debugging info.
−−input is used for taking input file as already used above.
−−output is used for getting the required output as already shown above.

Note above that double dash are without any spaces. Also the tests folder contains the 10 sample JAVA programs as required.

The Extra Functionalities Implemented are atleast :
1) Strings
2) Multidimensional Arrays (which are greater than 3 dimesnions as well)
3) Constructor
4) Type Casting