CS335 Project Milestone 1

Akhil Jain
200077
akhilj20@iitk.ac.in

Harsh Jain
200412
harshj20@iitk.ac.in

Jasjot Singh
200468
jasjots20@iitk.ac.in

1 Introduction

We used the **Flex lexical analyzer** for generating tokens from the input Java program. We used the **Bison parser generator** for generating a parser that can parse Java code and then convert it into a DOT file which can be used to view the **Abstract Syntax Tree** (AST). This AST can be visualized if you have Graphviz installed. The compilation instructions include the following:

- flex lexer.l
- bison -d -v -t parser.y
- g++ lex.yy.c parser.tab.c

However, all the compilation instructions have instead been put together in a makefile. You can run this by simply calling the command:

- make

This should now generate an output file named **myASTGenerator** and now you can execute it as follows:

- ./myASTGenerator <input-filename> <output-filename>

Another way to run this is as follows:

- ./myASTGenerator −−input=FILE −−output=FILE

By default, the first argument after the "./myASTGenerator" will be treated as the input filename and the second output will be the output filename.

If only one argument is mentioned after the "./myASTGenerator", then that argument is taken to be the input file and the output file is named **parse_tree.dot** by default.

You can use the following command to view the helper page:

- ./myASTGenerator −−help

For debugging suggestions in case of error, use the following command:

- ./myASTGenerator <input-filename> −−verbose

Once the DOT file has been generated, the following command generates a **.ps** file that can be used to view the AST.

- dot −Tps <dot-file-name> −o < Visual>.ps

Support for the following functionalities has been added as optional features:

- Support for **Interfaces**
- **import** statements
- Type Casting
- Static Polymorphism
- Dynamic Polymorphism
- Any other features - **enum** support

These are the assumptions made in the first milestone of the project:

- Annotations are ignored
- We have ignored <> operator and related non terminals have also been ignored
- While declaring arrays, the size is not allowed to be predefined.
  For example:
  int a[ ] is valid, but int a[3] is not.
- The lexeme of text block is assumed to be "**TextBlock**" and its token is "**Literal**".
- The commands −− verbose should be written after mentioning all the input and output filenames