CS335A
Project
Milestone 2

Ashok Vishwakarma(180151)
Viraj Ravjibhai Limbasiya (180870)

24 March 2023

Tools used

We implemented the symbol table using a **nested dictionary** data structure in Python. It supports methods like add variable and search variable, etc.
The outer dictionary contains the scopes, and the inner dictionaries contain the identifier information for each scope.
We implemented the 3AC using an in-memory data structure (List) in python. Finally, we dump the 3-address code (3AC) to the console in the txt format.
We used the output format as [des, src1, src2, op], i.e., destination, source1, source2, operator for dumping the 3AC.
Example: For $a = b + c \rightarrow [a, b, c, +]$
The software/utilities used are:

- **Ply (Python Lex-Yacc):** It is a Python implementation of the popular Lex and Yacc tools for parsing and analyzing complex languages. It provides a set of tools for generating lexer and parser code in Python from a set of grammar rules specified by the user.
- **ply.lex** is a lexer generator that takes in a set of regular expression patterns and generates a lexer in Python code.
- **ply.yacc** is a parser generator that takes in a set of grammar rules and generates a parser in Python code.

Build/Execute

The Lexical grammar for Java8 is present in JavaLexer.py which has been obtained from Java 17 Lexical Structure.
The Syntactic actions for grammar are present in IRGen.py which has been obtained from Java 17 Syntax.

To run the source code, use the following command:

> make

which internally calls ‘ python3 ./IRGen.py ../tests/test.java‘
It generates a symbol table as a CSV file of input Java file and also generates the 3AC in txt file.