CSCI 447 – Semantic Analysis
Professor: Ahmed Rafea

Objectives:
To augment the recursive decent parser or the bottom up parser for the C- language you have developed in the parsers assignments to:.
- store variables and their attributes in a symbol table
- handle semantic errors.
- generate a three address code as an intermediate representation.

C- Language
The same specifications described in assignment 2

Symbol Tables:
All declared identifiers should be entered in symbol tables. A symbol entry for a variable should store its name, its category (simple , array), its type (in case of an array, it will be the type of elements), a field that contains the number of elements in case of an array type(contains **o** in case of simple type), its relative address, and its declaration line number in the source file. Hashing techniques should be used to speed up the insert and lookup functions. You can assume that, integer occupies 2 bytes and float occupies 4 bytes.

Static Semantics Checking:
The following static semantic rules should be checked. Error messages should be reported when these rules are violated, but parsing and translation should continue. Error message should specify the line number and describe the error that occurred.

1. All variables must be declared before they can be used.
2. Operands of +,-,*,/ must be integer or float
3. No mixed types are permitted in any operation or assignment

Translation into an Intermediate Representation (Bonus)
A three address code is the intermediate representation for this assignment. The three address code can be implemented as a linked list of quadruples where operands of the three address code are either pointers to symbol table, or numeric constants

Output the Semantic Analyzer:
For each variable: output its name, its type, its relative address and its line number and a list of the three address code in case you implement the bonus part.

Report:
A detailed report should be written explaining your design and implementation. Specifically, you need to discuss:
- The attribute grammar including semantic rules and actions
- The symbol tables structure
- Implementation Issues and decisions
- Handling semantic errors, and
- Sample Runs of Test cases

To submit by email:
1. The report document.
2. All source files and executable file

Grading

Your grade will be divided into the following components:
1. Correctness and output
2. Attribute Grammar completeness and correctness
3. Symbol table, implementation details and documentation
4. Handling semantic errors, error reporting
5. Report Document