Compiler Project Documentation

Students: Arlind Lacej, Ridvan Plluzhina

Subject: Formal Languages and Compilers

Academic Year: Second Semester 2024/25

1. What the Compiler Does

This project implements a compiler for a simple programming language using FLEX (lexer) and YACC/Bison (parser). The compiler supports variable declarations, assignments, arithmetic expressions, and basic type checking. It allows two variable types: int and float. The compiler checks for undeclared variables and type mismatches, evaluates arithmetic expressions, and reports semantic errors with clear messages.

2. Grammar of the Language

The language grammar supports:
- Declarations of int and float variables
- Assignments
- Arithmetic expressions using +, -, *, /

Grammar rules:
program → program stmt | ε
stmt → decl ';' | assign ';'
decl → INT_TYPE ID | FLOAT_TYPE ID
assign → ID '=' expr
expr → expr '+' expr | expr '-' expr | expr '*' expr | expr '/' expr | NUM | ID

3. What the Input Should Look Like

The input consists of code written in the defined language. Each statement must end with a semicolon. Variables must be declared before they are used in assignments or expressions. Valid inputs include:
- int x;
- float y;
- x = 5;
- y = x + 3.5;

4. How to Run the Compiler

To compile and run the compiler using the MSYS2 terminal:

1. Run `flex compiler.l`
2. Run `bison -d parser.y`
3. Compile using: `gcc parser.tab.c lex.yy.c symbol_table.c -o my_compiler`
4. Run with: `./my_compiler < tests/test1.txt`

Ensure the input files (e.g., test1.txt) are located in the 'tests' directory.