Compilers Project Report (CMPN403)

Omar Alaa (1190377)   Omar Nashat (1190430)
Rana Gamal (1190449)

May 17, 2024

Project Overview:

This project is a compiler for a custom language that is a subset of C. The compiler is written in C++ and uses flex and bison for lexical and syntax analysis. The compiler generates quads and symbol tables for the input code. The compiler also has a GUI written in Python using Tkinter. The GUI allows the user to write or import code in the custom language and then compile it.

Tools and Technologies:

Compiler:

- **Language:** C++
- **Compilers:** g++, bison, flex
- **Build:** CMake

GUI:

- **Language:** Python
- **Framework:** Custom Tkinter

Tokens:

- **PROGRAM:** The whole program
- **STMT_LIST:** List of statements
- **BLOCK:** Block of code `{ ... }`
- **STMT_LIST_EPS:** Either empty or a list of statements
- **STMT:** A single statement
- **OPENSCOPE:** Open scope `{`
- **CLOSESCOPE:** Close scope `}`
- **NON_SCOPED_STMT:** Statement that is not in a scope
- **SCOPED_STMT:** Statement that is in a scope
- **DATA_TYPE:** Data type of a variable
- **VARIABLE:** Variable name
- **ASSIGN:** Assignment operator `=`
- **ASSIGN_OP:** Assignment operator `+=`, `-=`, `*=`, ...
- **EXPR:** Expression
- **CONST:** Constant value
- **INC:** Increment `++`
- **DEC:** Decrement `--`
- **MATH_OP:** Mathematical operator `+`, `-`, `*`, ...
- **BITWISE_OP:** Bitwise operator `&`, `|`, `^`, ...
- **BOOL_EXPR:** Boolean expression
- **DATA_LITERALS:** Data literals
- **INT_LITERAL:** Integer literal
- **FLOAT_LITERAL:** Float literal
- **CHAR_LITERAL:** Char literal
- **BOOL_LITERAL:** Boolean literal
- **LOGICAL_OP:** Logical operator `&&`, `||`, ...
- **IF_COND:** If condition
- **ELSE_TOK:** Else token
- **WHILE_TOK:** While token
- **WHILE_COND:** While condition
- **DO:** Do token
- **FOR_HEAD:** For head
- **FOR_STMT:** for init
- **FOR_COND:** for condition
- **SWITCH_HEAD:** Switch head
- **CASES:** Cases for switch
- **CASE_STMT:** Case statement
- **CASE_STMT_COND:** Case statement condition
- **FUNCTION:** Function
- **FUNCTION_START:** Function start
- **PARAMS:** Function parameters
- **PASSED_PARAMS:** Passed function parameters
- **RETURN:** Return statement

Quads:

ID	Operation	Description
0	`>`	Greater than
1	`<`	Less than
2	`>=`	Greater than or equal to
3	`<=`	Less than or equal to
4	`==`	Equal to
5	`!=`	Not equal to
6	`&&`	Logical AND
7	`||`	Logical OR
8	`&`	Bitwise AND
9	`|`	Bitwise OR
10	`ˆ`	Bitwise XOR
11	`~`	Bitwise NOT
12	`>>`	Bitwise right shift
13	`<<`	Bitwise left shift
14	`+`	Addition
15	`-`	Subtraction
16	`*`	Multiplication
17	`/`	Division
18	`%`	Modulus
19	`=`	Assignment
20	`+=`	Addition assignment
21	`-=`	Subtraction assignment
22	`*=`	Multiplication assignment
23	`/=`	Division assignment
24	`%=`	Modulus assignment
25	`++`	Increment
26	`--`	Decrement
27	`JMP`	Jump True
28	`JMPF`	Jump False
29	`SETLiteral`	Set Literal
30	`SETLabel`	Set Label
31	`CALL`	call function