Compiler project

Team 5

Submitted to:

Engr. Nesma Refaei

Submitted By:

Islam Ahmed
Mohamed Abo Bakr
Mohamed Ibrahim
Omar Tarek

Language supported

We made a language that is nearer to being like C++ than others. We added many features like:

- Variables could be initialized at the time of declaration
- Variables could be (int, double, bool, char, string)
- const typed variables are supported
- Many control structures
    - If-else
    - If-elif-else
    - While
    - For
    - Repeat-until
    - Switch-case-default
- Most Logical and Mathematical operators

In case of either Syntax error or Semantic error:

- Symbol table is printed
- Quadruples are not printed
- Errors are printed

Symbol table

We keep track of all the variables and their status like:

- Var_name
- Var_type
- Is_constant
- Is_initialized

- Is_used
- Var_line_num

Quadruples

After checking what's actually needed to implement whether Assembly or Three Adress Code (TAC) and knowing that TAC is intended. We've implemented a typical TAC and then transferred it to the Quadruples table.

Example				Description
Op	arg1	arg2	dst	
JMP			label1	Its unconditional jump to label1
JNE	true	x	label1	Its conditional jump to label1 if x not equal true
"="	x		t0	Its copying x value to the temporary variable t0
"+"	x	5	t1	Its adding two values: X and 5. Then move it to the temporary variable t1

List of tokens

"if"
"elif"
"else"
"while"
"for"
"switch"
"case"
"default”
"repeat"
"until"
"end"
"int"
"double"
"bool"
"char"
"string"
"const"
"&&"
"||"
"=="
"!="
">"
">="
"<="
"<"
"-"
"+"
"*"
"/"
"="
"false"
"true"
":"
";"
“VARNAME”
“DIGIT”
“DOUBLE”
“CHAR”
“STRING

Project Structure

There are few files to avoid overhead having the project core.

- Scanner.l
  - Contain the regex and used by lex
- Parser.y
  - Contain the grammar rules and used by yacc.
- Test.txt
  - Its the code file to be compiled

How To Run

1. Prerequisites:
    a. Install bison (GNU Bison) v 3.8.2
    b. Install flex v 2.6.4
2. Open the terminal in project directory
3. Run these 3 lines to generate the .exe file
    a. flex ./scanner.l
    b. bison -d ./parser.y
    c. g++ -o test.exe lex.yy.c parser.tab.c
4. Now you're ready to run the project using this command ./test.exe ./Test.txt