Assignment-12

33119

Title: LEX and YACC program to generate intermediate code.

Program Statement: Write a program for intermediate code generation using LEX and YACC for Control Flow Statement. (Either while loop or Switch case).

Objective:

- To understand forth phase of compiler: intermediate code generation.
- To learn and use compiler writing tools.
- To learn how to write three address code for given statement.

Theory:

Introduction:

In the analysis-synthesis model of a compiler, the front end analyzes a source program & creates an intermediate representation, from which the back end generates target code. Ideally details of the source language are confined to the front end, and details of the target machine to the back end.

Intermediate Languages:

Three ways of intermediate representation

- Syntax tree.
- Postfix notation
- Three address code.

Steps to execute the program

1. $ lex filename.l
2. $ yacc -d filename.y
3. $ cc lex.yy.c y.tab.c -ll -ly -lm
4. $ ./a.out
5. (eg. compl)
6. (eg. comp-4)

Algorithm :

Write a LEX & YACC program to generate Intermediate code for arithmetic expression

LEX program.

1. Declaration of header files specially y.tab.h which contains declaration for letter, Digit, expr.
2. End declaration section by %%
3. Match regular expression.
4. If match found then convert it into char & store it in yylval.p where p is pointer declared in YACC.
5. Return token.
6. If input contains new line character (\n) then return 0.
7. If input contains "." then return yytext[0]
8. End rule-action section by %%
9. Declare main function.
10. a. Open file given at command line.
11. b. If any error occurs the print error.
12. c assign file pointer fp to yyin.
13. d call function yylex until file ends.
14. End.

YACC Program :

1. Declaration of header file.
2. Declare structure for three adress code repres-entation having fields of argument 1. argument 2. operator result.
3. Declare pointer of char type in union.
4. Declare token expr of type pointer p.
5. Give preedence to "*", "/"
6. Give precedence to "+", "-"
7. End of declaration section by %%
8. If final expression evaluates then add it to the table of three address code.
9. If input type is expression of the form.
10. a. exp "+" exp then add to table the argum argument 2 operator.
11. b. exp "-" exp then add to table the argument 1 argument 2 operator
12. c. exp "*" exp then add to table the argument 1, argument 2 operator
13. d exp "/" exp then add to take the argument 1. argument 2 operator.
14. e. "(" exp ")" then assign $2 to $$
15. f. Digit OR Letter then assign $1 to $$
16. End the section by %%
17. Declare file * yyin externally.
18. Declare main function & call yy parse function with yyin ends.
19. Declare yyerror for if any error occurs.
20. Declare char pointer s to print error

21. Print error message.
22. End of program.

Conclusion : Thus, I have successfully implement program for intermediate code generation using LEX & YACC for control flow.