Name - Vasu Kalariya
Roll - PE29
Sub - SSC

Lab Assignment - 7

Title: Validation of ~~comparer~~ compound statement

Problem statement: write a program using LEX & YACC to validate compound statements in High Level Language

Objective:
→ To study YACC tool for syntax analysis
→ To master YACC utility

Theory:

Syntax Analysis of a Parser
Syntax ~~to~~ Anysising is second phase of the compiler design that comes after lexical analysis. It analyses the syntatical structure of the given input. It checks if the given input is in the correct syntax of the programing language in which the input which has been written. It is known as the Parse Tree ~~of~~ or Syntax Tree

Definitions of yylval, yyparse() & yyerror()

yylval: tab h1. generated by yacc with the -d option. This variable is defined as a C union having a number function called text to point to character strings and a member val to hold a integer value. This defination was performed in the yacc specification
yyparse(): You call this function to cause parsing to occur. This function reads tokens, executes actions and ultimately returns when it encounters end-of-input.

yyerror: It is a lex and yacc function that simply display a text using string argument to stderr using fprintf, and returns the integer value recieved from fprint.

Input: Source specification (*.y) file for loop statement like if statements while and do-while statement

Output: Statement is gramatically correct or not

Conclusion: Parser for validation of compound statement is successfully done.

FAQ's

1 What is the role of y.tab.h file?

Before writing a LEX program, there must be some way by which the YACC program can tell the LEX declared in the YACC program. This a facilitated by the file y.tab.h which contains the file

Eg: # define DIGIT 253

Note that 253 is a YACC generated constant to represent DIGIT. The constant may vary at different defining a macro identifier corresponding to it.

2 Explain YACC tool.

YACC stands for Yet Another compiler. It provides a tool to produce a parser for a given grammer and it designed to compile a LALR (1) grammer

YACC is used to produce a source code of the syntactic analyzer of the language produced by LALR (1) grammer the input of YACC is the rule or grammer and the output is a C program.

```
%{
#include "sample.tab.h"
extern int yyerror(char *str);
extern int yyparse();
%}
%%
"while" return WH;
"if" return IF;
"do" return DO;
"for" return FOR;
"(" return OP;
")" return CP;
"{" return OCB;
"}" return CCB;
"<" |
">" |
"<=" |
">=" |
"==" |
"!=" return CMP;
"+" |
"-" |
"*" |
"/" return OPR;
"=" return ASG;
([a-zA-Z])("_"|[a-zA-Z0-9])* return ID;
[0-9]+ return NUM;
";" return SC;
"," return COMMA;
" " {}
%%
int yywrap()
{
return 1;
}

%{
#include<stdio.h>
extern int yylex();
extern int yywrap();
extern int yyparse();
%}
%token WH IF DO FOR OP CP OCB CCB CMP SC ASG ID NUM COMMA OPR
%%

start: swh | mwh | dowh | sif | mif;
swh: WH OP cmplst CP stmt {printf("VALID SINGLE STATEMENT WHILE LOOP\n");};
mwh: WH OP cmplst CP OCB stlst CCB {printf("VALID MULTI STATEMENT WHILE LOOP\n");};
dowh: DO OCB stlst CCB WH OP cmplst CP SC {printf("VALID DO-WHILE LOOP\n");};
sif: IF OP cmplst CP stmt {printf("VALID SINGLE STATEMENT IF\n");};
```

```
mif: IF OP cmplst CP OCB stlst CCB {printf("VALID MULTI STATEMENT IF\n");};
cmplst: cmpn COMMA cmplst | cmpn ;
cmpn: ID CMP ID | ID CMP NUM;
stlst: stmt stlst | stmt;
stmt: ID ASG ID OPR ID SC | ID ASG ID OPR NUM SC | ID ASG NUM OPR ID SC | ID ASG NUM
OPR
NUM SC | ID ASG ID SC | ID ASG NUM SC
| start {printf("NESTED INSIDE A ");}
;
%%
int yyerror(char *str)
{
printf("%s", str);
}
main()
{
yyparse();
}
```