**✅ Title: Validation of Compound Statement Using LEX and YACC Tool**

**🎯 Aim:**

To write a program using **LEX and YACC** to validate compound statements in high-level languages (like if, while, do-while).

**📌 Objectives:**

1. To study **syntax analysis** using YACC tools.
2. To understand **YACC utilities** for grammar rule implementation.

**📖 Theory Summary**

**🧩 1. Syntax Analysis Phase (Parsing):**

* Second phase of a compiler after lexical analysis.
* Checks whether the given token sequence follows the **grammar** of the language.
* Builds **parse trees** or **abstract syntax trees (ASTs)**.

**🧰 2. Inbuilt Variables & Functions:**

| **Component** | **Purpose** |
| --- | --- |
| **yylval** | Used to pass values from LEX to YACC. |
| **yyparse()** | Main parsing function generated by YACC. |
| **yyerror()** | Error-handling function for syntax errors. |

**⚙️ 3. Compilation & Execution Process**

1. Write .l file for LEX and .y file for YACC.
2. Run:
3. yacc -d file.y
4. lex file.l
5. gcc lex.yy.c y.tab.c -o output
6. ./output
7. Input: Compound statement (e.g., if(a > b){ x = 1; })
8. Output: Grammar validation message (Success or Syntax Error)

**🧪 Input:**

* Valid compound statements like if-else, while, do-while, blocks with {}.

**📤 Output:**

* Output indicates whether the **syntax is valid or invalid**.

**✅ Conclusion:**

The parser for **compound statement validation** was successfully implemented using **LEX and YACC**.

**💻 Platform:**

Linux (LEX & YACC)

**🎤 Viva Questions & One-Line Answers**

**🧠 Concepts of YACC & Parsing**

1. **What is the role of YACC in a compiler?**  
   → YACC is used for syntax analysis by implementing grammar rules.
2. **What is a compound statement?**  
   → A block of code enclosed in {} that can contain multiple statements.
3. **What is the format of a .y file in YACC?**  
   → It has three sections: definitions, rules (grammar), and user code.

**🔧 Functions & Execution**

1. **What does yyparse() do?**  
   → It invokes the parser and starts syntax analysis.
2. **What is yyerror() used for?**  
   → It is called when a syntax error is encountered.
3. **What is yylval?**  
   → A variable used to pass data from LEX to YACC.
4. **What does y.tab.h contain?**  
   → It contains token definitions generated by YACC.
5. **What are tokens in YACC?**  
   → Tokens are terminal symbols passed from LEX (like IF, WHILE, ID).

**🧪 Testing & Output**

1. **What is the output for a valid compound statement?**  
   → It prints "Valid syntax" or similar success message.
2. **What happens if there's a syntax error?**  
   → yyerror() prints an error message and parsing stops.
3. **How do LEX and YACC work together?**  
   → LEX handles tokens, YACC parses them according to grammar rules.

**Compound.l**

%{

#include "y.tab.h"

extern int yyerror(char \*str);

extern int yyparse();

%}

%%

"if" return IF;

"while" return WH;

"(" return OP;

")" return CP;

"<" |

">" |

"<=" |

">=" |

"==" |

"!=" return CMP;

"+" |

"-" |

"\*" |

"/" return OPR;

"=" return ASG;

([a-zA-Z])("\_"|[a-zA-Z0-9])\* return ID;

[0-9]+ return NUM;

";" return SC;

" " {}

%%

int yywrap() {

return 1;

}

**Compound.y**

%{

#include<stdio.h>

extern int yylex();

extern int yywrap();

extern int yyparse();

extern int yyerror(char \*str);

%}

%token WH IF DO FOR OP CP OCB CCB CMP SC ASG ID NUM COMMA OPR

%%

start: sif | swhile |sfor;

swhile: WH OP cmpn CP stmt {printf("VALID STATEMENT WHILE\n");};

sfor: FOR OP stmt cmpn SC stmt CP stmt {printf("VALID STATEMENT FOR\n");};

sif: IF OP cmpn CP stmt {printf("VALID STATEMENT IF\n");};

cmpn: ID CMP ID | ID CMP NUM;

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;

%%

int yyerror(char \*str)

{

printf("%s", str);

}

int main()

{

yyparse();

return 1;

}