

Roll no : PD-05

Aniruddha Shende

Panel : D



Dr. Vishwanath Karad

**MIT WORLD PEACE
UNIVERSITY** | PUNE

TECHNOLOGY, RESEARCH, SOCIAL INNOVATION & PARTNERSHIPS

T.Y.B.Tech (CSE)

System Software and Compilers(SSC)

Lab Assignment No – 7

Name: Aniruddha Shende

Roll number: PD-05

Batch: D1

Panel: D

Name:- Aniruddha Arun Shende

Roll no:- PD05

Batch:- DI

Panel :- D



Subject :- SSC

Lab Assignment No- 7

Validation of Control Stmt

Title:- Validation of compound stmt using LEX & YACC tool:

Problem Statement:- Write a program using LEX & YACC to validate compound statements in High Level Language.

Objective:-

- ① To study YACC tools for syntax analysis.
- ② To master YACC utility.

Theory:-

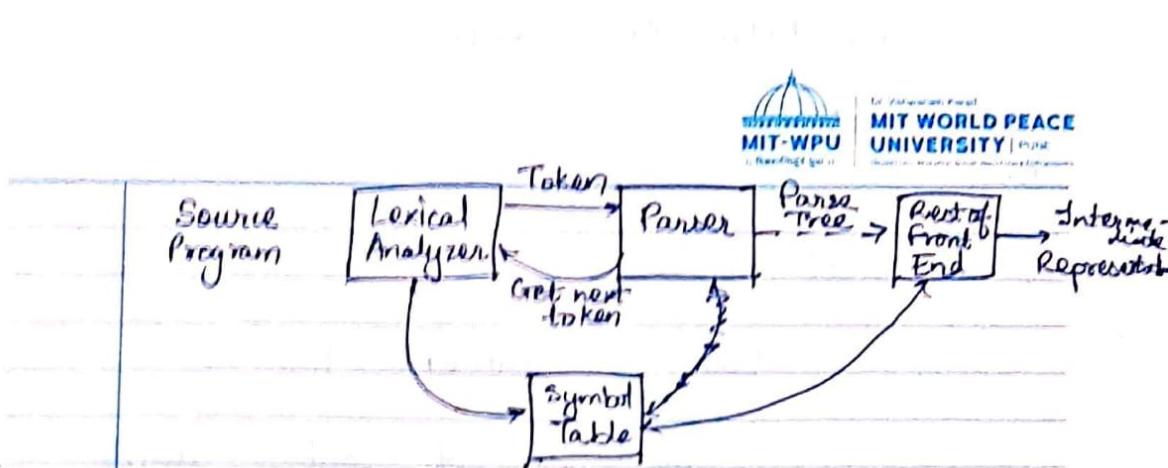
1. Write about the syntax analysis phase of the compiler.

Ans:- The main objective of syntax analysis phase is checking whether the token stream meets the grammatical specification of language & generates the syntax tree.



Scanned with
CamScanner

www.mitwpu.edu.in



2. Description of Each section of *.y file with example-

Ans.2 YACC input file is saved with *.y extension.

It contains 3 sections:

```
/* definitions */ ....
% . % . rules %
% . %
/* auxiliary routines */
```

Definition Part: It includes information about the tokens used in syntax definition.

Eg:- % token NUMBER 627

Rules Part: It includes information about the tokens used in the syntax definition, grammar definition in a modified BNF form actions in C code in {} & can be embedded inside {Translation Schemes}.

Auxiliary Routines Part: It is only C code. It include function definitions for every function needed in rules part. It can also main () function definition.

3. Description of standard inbuilt variables & functions like yyval, yyparse(), yyerror().

- Ans. 3.
- (a) `yyval()`: This function gives value associated with that token.
 - (b) `yyparse()`: It returns a value 0 if the input it parser is valid according to the given grammar rules. It return value 1 if the input is incorrect & error recovery is impossible.
 - (c) `yyerror()`: The YACC passes one argument to this function: a character string describing the type of error that just took place.

4. Compilation & Execution Process.

Ans. 4. Compilation:-

```
flex prog.l
bison -dy prog.y
gcc lex.yy.c y.tab.c
```

Execution :-

- ./a.out.



Input:- Source specification (*.y) file for loop stmt like if statements, while & do-while stmts.

Output:- Parser for validation of compound statement is successfully done.

Platform:- Linux (Lex & YACC)

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

FAQ's:-

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

Ans.1. This file is generated which contains definitions of token names.

2. Explain YACC tool.

Ans.2. YACC stands for Yet Another Compiler Compiler. YACC provides a tool to produce a parser for a given grammar. It is used to produce the source code of the syntactic analyzer of the language produced by LALR(1) grammars. YACC is a program designed to compile a LALR(1) grammar.



Code & Output:**lab7.l :**

```
%
#include "y.tab.h"
extern int yyerror(char *str);
extern int yyparser();
%}

%%

"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;
}
```

lab7.y :

```
%
#include<stdio.h>
extern int yylex();
extern int yywrap();
extern int yyparse();
%}

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

%%
start: swh | mwh | dowh | sif | mif;
swh: WH OP cmpn CP stmt
    {printf("VALID SINGLE STATEMENT
WHILE LOOP\n");};
```

```

mwh: WH OP cmpn CP OCB stmt CCB           {printf("VALID MULTI STATEMENT
WHILE LOOP\n");};

dowh: DO OCB stmt CCB WH OP cmpn CP SC    {printf("VALID DO-WHILE
LOOP\n");};

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

mif: OF OP cmpn CP OCB stmt CCB          {printf("VALID MULTIPLE
STATEMENT IF");};

cnob: ID CMP ID | ID CMP NUM;

stmt: ID ASG ID OPR ID SC | ID ASG ID OPR NUM SC | ID ADG NUM OPR ID SC | ID
ADG NUM OPR NUM SC | ID ASG NUM | start {printf("NESTED INSIDE A ");}

%%

int yyerror(char *str) {
    printf("%s", str);
}

main() {
    yyparse();
}

```

Output:

```

Terminal
student@admin:~/Desktop
student@admin:~$ cd Desktop/
student@admin:~/Desktop$ ls
lab7.l lab7.y
student@admin:~/Desktop$ flex lab7.l
student@admin:~/Desktop$ byacc -b lab7.y
Usage: byacc [options] filename
      Options:
        -b file_prefix      set filename prefix (default "y.")
        -B                  create a backtracking parser
        -d                  write definitions (.tab.h)
        -i                  write interface (y.tab.i)
        -g                  write a graphical description
        -l                  suppress #line directives
        -L                  enable position processing, e.g., "%locations"
                           (default ".tab.c")
        -o output_file       set symbol prefix (default "yy")
        -p symbol_prefix    create a reentrant parser, e.g., "%pure-parser"
        -P                  produce separate code and table files (y.code.c)
        -r                  suppress #define's for quoted names in %token lines
        -s                  add debugging support
        -t                  write description (y.output)
        -v                  show version information and exit
student@admin:~/Desktop$ byacc -d lab7.y
student@admin:~/Desktop$ gcc lex.yy.c y.tab.c
student@admin:~/Desktop$ ./a.out
if(x == a)c = 14;
VALID SINGLE STATEMENT IF

```