Roll No: 20BCE204
Course Name and Course Code: 2CS701 Compiler Construction
Practical No: 5
Aim: To implement a calculator in YACC: Syntax Directed Translation

Extend practical assignment 1 to generate a Symbol Table for identifiers, and label in the code. (Symbol Table columns: Name, Value)

Use YACC to Write a Grammar for multiple expression statements, and apply syntax directed translation for calculator.

Code:

5a.l

```
%{
/* Definition section */
#include<stdio.h>
#include "y.tab.h"
%}

/* Rule Section */
%%
[0-9]+ {
                    yylval=atoi(yytext);
                    return NUMBER;

            }
[\t] ;

[\n] return 0;

. return yytext[0];

%%

int yywrap()
{
return 1;
}
```

5a.y

```
%{
/* Definition section */
#include<stdio.h>
int flag=0;
extern int yylex();
void yyerror(const char* msg);
%}

%token NUMBER

%left '+' '-'

%left '*' '/' '%'
```

```
%left '(' ')'

/* Rule Section */
%%

ArithmeticExpression: E{

printf("\nResult=%d\n", $$);

return 0;

};
E:E'+'E {$$=$1+$3;}

|E'-'E {$$=$1-$3;}

|E'*'E {$$=$1*$3;}

|E'/'E {$$=$1/$3;}

|E'%'E {$$=$1%$3;}

|'('E')' {$$=$2;}

| NUMBER {$$=$1;}

;

%%

//driver code
void main()
{
printf("\nEnter Expression:\n");

yyparse();
if(flag==0)
printf("\nExpression is Valid Expression\n\n");
}

void yyerror(const char* msg)
{
printf("\nExpression is Invalid Expression\n\n");
flag=1;
}
```

5b.1

```
%{
/* Definition section */
#include<stdio.h>
#include "y.tab.h"

%}

/* Rule Section */
%%
[a-zA-Z] {
```

```
    return LETTER;
}

[0-9]+ {

    return DIGIT;
}

[\t] ;

[\n] return 0;

. return yytext[0];
%%

int yywrap()
{
    return 1;
}
```

5b.y

```
%{

#include<stdio.h>
int flag = 0;
extern int yylex();
void yyerror(const char* msg);
%}

%token DIGIT
%token LETTER EPS


%%
ArithmeticExpression: S {
    printf("\nResult=%d\n", $1);
}
;

S: L A|;
A: M A| D A|;
L: LETTER;
M: LETTER;
D: DIGIT;

%%

int main()
{
    printf("\nEnter the string:\n");
    yyparse();
    if (flag == 0)
        printf("\nEntered String is Valid\n\n");
    return 0;
}

void yyerror(const char* msg)
{
```

```
    printf("\nEntered String is Invalid\n\n");
    flag = 1;
}
```

Output: