1. **Introduction to YACC tool**.

YACC

- YACC stands for **Yet Another Compiler Compiler**.
- YACC provides a tool to produce a parser for a given grammar.
- YACC is a program designed to compile a LALR (1) grammar.
- It is used to produce the source code of the syntactic analyzer of the language produced by LALR (1) grammar.
- The input of YACC is the rule or grammar and the output is a C program.

These are some points about YACC:

Input: A CFG- file.y

Output: A parser y.tab.c (yacc)

- The output file "file.output" contains the parsing tables.
- The file "file.tab.h" contains declarations.
- The parser called the yyparse ().
- Parser expects to use a function called yylex () to get tokens.

2. **Use YACC tool to generate Calculator Program**

Code:

calc.l

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

#include "y.tab.h"
extern int yylval;
%}

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

    }
```

```
[\t] ;

[\n] return 0;

. return yytext[0];

%%

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

Calc.y

```
%{

/* Definition section */

#include<stdio.h>

int flag=0;

%}


%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 Any Arithmetic Expression which can have operations Addition,Subtraction,
Multiplication, Division,          Modulus and Round brackets:\n");

yyparse();
if(flag==0)
printf("\nEntered arithmetic expression is Valid\n\n");
}
```

```
void yyerror()
{
printf("\nEntered arithmetic expression is Invalid\n\n");
flag=1;
}
```

Output: