LAB ASSIGNMENT -3

YACC TOOL

Name: Sharan P **Reg.No:** 21BRS1582

PROGRAM 1.

Aim: To generate yacc specification for a program to recognize a valid arithmetic expression that uses operator +,-,*,/

Code:

LEX CODE:

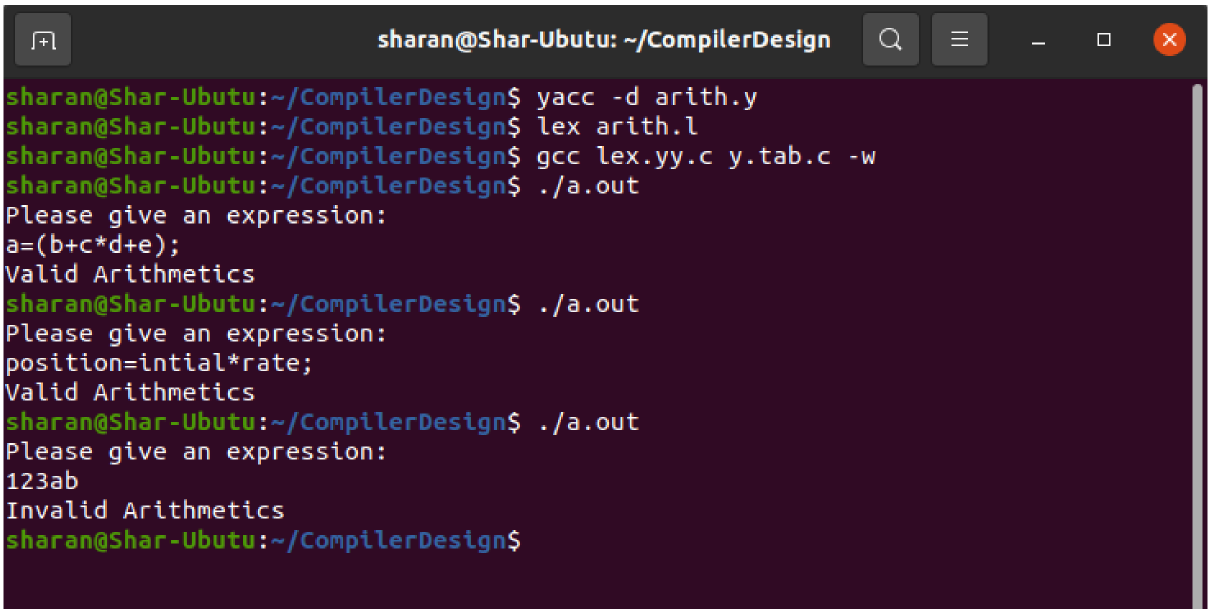
```
%{
#include"y.tab.h"
%}

%%

[a-zA-Z_][a-zA-Z_0-9]* return id;
[0-9]+(\.[0-9]*)?      return num;
[+/*]                  return op;
.                     return yytext[0];
\n                     return 0;

%%

int yywrap() {
          return 0;
}
```

YACC CODE:

```
%{
   #include<stdio.h>
   int valid=1;
%}

%token num id op

%%

start : id '=' s ';'
```

```
s :   id x      | num x      | '-' num x    | '(' s ')' x ;
x :   op s      | '-' s      | ;

%%

int yyerror(){
   valid=0;
   printf("Invalid Arithmetics\n");
   return 0;
}

int main(){
   printf("Please give an expression:\n");
   yyparse();
   if(valid)
   {
      printf("Valid Arithmetics\n");
   }
}
```

Output Screenshot:



```
sharan@Shar-Ubutu:~/CompilerDesign$ yacc -d arith.y
sharan@Shar-Ubutu:~/CompilerDesign$ lex arith.l
sharan@Shar-Ubutu:~/CompilerDesign$ gcc lex.yy.c y.tab.c -w
sharan@Shar-Ubutu:~/CompilerDesign$ ./a.out
Please give an expression:
a=(b+c*d+e);
Valid Arithmetics
sharan@Shar-Ubutu:~/CompilerDesign$ ./a.out
Please give an expression:
position=intial*rate;
Valid Arithmetics
sharan@Shar-Ubutu:~/CompilerDesign$ ./a.out
Please give an expression:
123ab
Invalid Arithmetics
sharan@Shar-Ubutu:~/CompilerDesign$
```

Result:

Using yaac tool we have created a program that recognizes a valid arithmetic expression using the arithmetic operator +,-,*,/

END

PROGRAM 2.

Aim: To generate yacc specification for a program to recognize a valid variable which starts with a letter followed by any number of digits or letters.

Code:

LEX CODE:

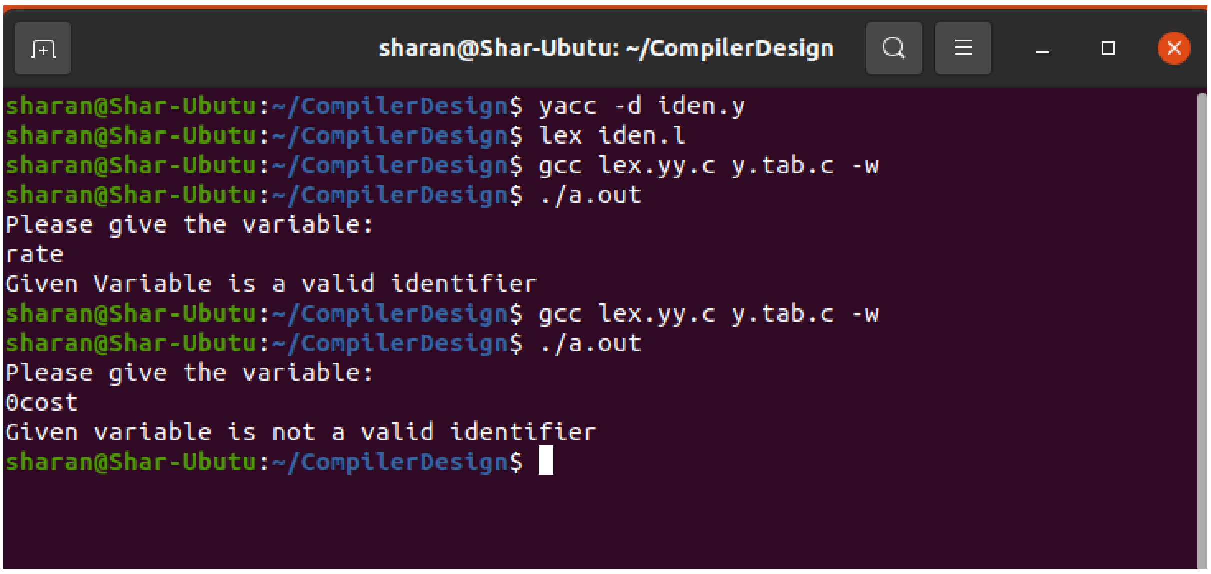
```
%{
#include"y.tab.h"
%}

%%

[a-zA-Z] {return LETTER;}
[0-9] {return DIGIT;}
[_] {return UND;}
[\n] {return NL;}
. {return yytext[0];}
%%

int yywrap() {
        return 0;
}
```

YACC CODE:

```
%{
#include<stdio.h>
#include<stdlib.h>
%}

%token DIGIT LETTER UND NL

%%
stmt: variable NL {printf("Given Variable is a valid identifier\n"); exit(0);}
;

variable: LETTER alphanumeric
;

alphanumeric: LETTER alphanumeric | DIGIT alphanumeric | UND alphanumeric | LETTER | DIGIT |
UND
;
```

```
%%

int yyerror(char *msg){
printf("Given variable is not a valid identifier\n");
exit(0);
}

main(){
        printf("Please give the variable: \n");
        yyparse();
}
```

Output Screenshot:



```
sharan@Shar-Ubutu:~/CompilerDesign$ yacc -d iden.y
sharan@Shar-Ubutu:~/CompilerDesign$ lex iden.l
sharan@Shar-Ubutu:~/CompilerDesign$ gcc lex.yy.c y.tab.c -w
sharan@Shar-Ubutu:~/CompilerDesign$ ./a.out
Please give the variable:
rate
Given Variable is a valid identifier
sharan@Shar-Ubutu:~/CompilerDesign$ gcc lex.yy.c y.tab.c -w
sharan@Shar-Ubutu:~/CompilerDesign$ ./a.out
Please give the variable:
0cost
Given variable is not a valid identifier
sharan@Shar-Ubutu:~/CompilerDesign$
```

Result:

Using yaac tool we have created a program that recognizes a valid variable which starts with a letter followed by any number of digits or letters.

END

PROGRAM 3.

Aim: To generate yacc specification for a program to implement a arithmetic calculator using lex and yaac.

Code:

LEX CODE:

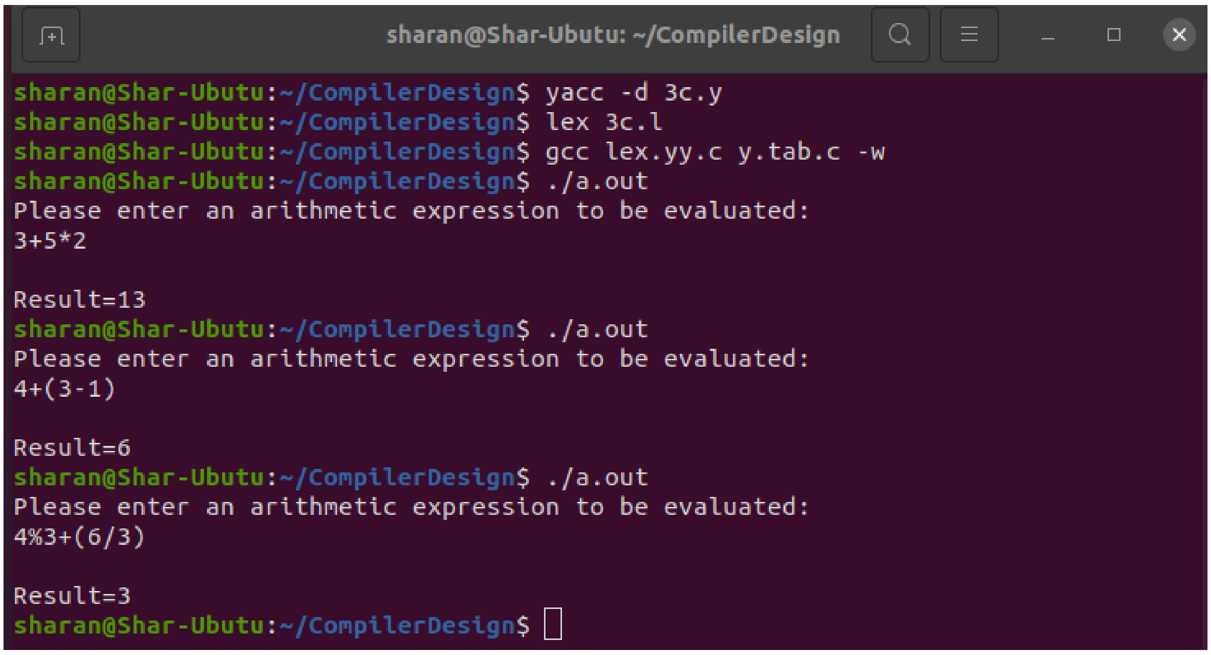
```
%{
#include<stdio.h>
#include"y.tab.h"
%}

%%

[0-9]+ {
                yylval = atoi(yytext);
                return NUMBER;

        }
[\t] ;

[\n] return 0;

. return yytext[0];

%%

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

YACC CODE:

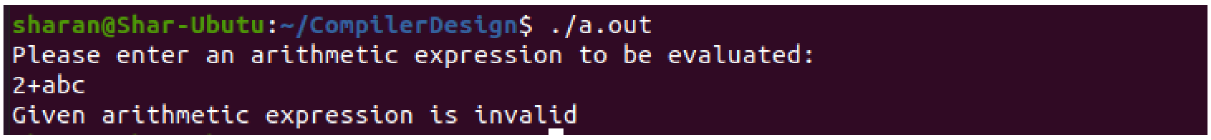
```
%{
#include<stdio.h>
#include<stdlib.h>
%}

%token NUMBER

%left '+' '-'

%left '*' '/' '%'

%left '(' ')'

%%
```

```
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;}

  ;

%%



int yyerror(char *msg){
        printf("Given arithmetic expression is invalid\n");
        exit(0);
}

main(){
        printf("Please enter an arithmetic expression to be evaluated: \n");
        yyparse();
}
```

Output Screenshot:



```
sharan@Shar-Ubutu:~/CompilerDesign$ yacc -d 3c.y
sharan@Shar-Ubutu:~/CompilerDesign$ lex 3c.l
sharan@Shar-Ubutu:~/CompilerDesign$ gcc lex.yy.c y.tab.c -w
sharan@Shar-Ubutu:~/CompilerDesign$ ./a.out
Please enter an arithmetic expression to be evaluated:
3+5*2

Result=13
sharan@Shar-Ubutu:~/CompilerDesign$ ./a.out
Please enter an arithmetic expression to be evaluated:
4+(3-1)

Result=6
sharan@Shar-Ubutu:~/CompilerDesign$ ./a.out
Please enter an arithmetic expression to be evaluated:
4%3+(6/3)

Result=3
sharan@Shar-Ubutu:~/CompilerDesign$ 
```



```
sharan@Shar-Ubutu:~/CompilerDesign$ ./a.out
Please enter an arithmetic expression to be evaluated:
2+abc
Given arithmetic expression is invalid
```

Result:

Using yaac tool we have created a program that implements an arithmetic calculator using lex and yaac.

END