Practical - 5

2CS701 – Compiler Construction

Harshit Gajipara
19BCE059

Aim:

To implement a calculator in YACC: Syntax Directed Translation Use YACC to Write a Grammar for multiple expression statements, and apply syntax directed translation for calculator.

Code:

Calculator.l

```
%{
        #include <stdio.h>
        #include "y.tab.h"
        extern int yylval;
%}

%%

[0-9]+ {yylval = atoi(yytext);
                return NUMBER;}
[\t] ;
[\n] return 0;
. return yytext[0];

%%
```

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

D:\19BCE059\B.Tech Semester 7\CC\CC Practicals\Practical 5\calculator.l - Sublime Text (UNREGISTERED)

File Edit Selection Find View Goto Tools Project Preferences Help

calculator.l

```
%{
    #include <stdio.h>
    #include "y.tab.h"
    extern int yylval;
%}

%%

[0-9]+ {yylval = atoi(yytext);
        return NUMBER;}
[\t] ;
[\n] return 0;
. return yytext[0];

%%

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

Line 1, Column 1

Calculator.y

```
%{
        #include <stdio.h>
        int flag=0;
%}

%token NUMBER
%left '+' '-'
%left '*' '/' '%'
%left '(' ')'

%%

Expression : 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;}
```

```
 | '('E')' {$$=$2;}
 | NUMBER  {$$=$1;}
 ;

%%

void main()
{
        while(1)
        {
                printf("\nEnter Arithmatic Expression : ");
                yyparse();
                if(flag==0)
                        printf("Valid Expression!\n");
        }
}

void yyerror()
{
        printf("Invalid Expression!\n");
        flag = 1;
}
```

D:\19BCE059\B.Tech Semester 7\CC\CC Practicals\Practical 5\calculator.y - Sublime Text (UNREGISTERED)

File Edit Selection Find View Goto Tools Project Preferences Help

calculator.l | calculator.y

```
%{
  #include <stdio.h>
  int flag=0;
%}

%token NUMBER
%left '+' '-'
%left '*' '/' '%'
%left '(' ')'

%%

Expression : 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;}
  | '('E')' {$$=$2;}
  | NUMBER  {$$=$1;}
 ;
```

Line 1, Column 1; Detect Indentation: Setting indentation to tabs with width 2

Output :

```
Activities    Terminal ▾                                    Oct 6  2:22 PM
                                          nirma@nirma-63: ~/Downloads/calculator
nirma@nirma-63:~/Downloads/calculator$ yacc -d  calculator.y
nirma@nirma-63:~/Downloads/calculator$ lex calculator.l
nirma@nirma-63:~/Downloads/calculator$ gcc lex.yy.c y.tab.c -ll
y.tab.c: In function ‘yyparse’:
y.tab.c:1221:16: warning: implicit declaration of function ‘yylex’ [-Wimplicit-function-declaration]
 1221 |       yychar = yylex ();
      |                ^~~~~
y.tab.c:1405:7: warning: implicit declaration of function ‘yyerror’; did you mean ‘yyerrok’? [-Wimplici
 1405 |       yyerror (YY_("syntax error"));
      |       ^~~~~~~
      |       yyerrok
calculator.y: At top level:
calculator.y:42:6: warning: conflicting types for ‘yyerror’
   42 | void yyerror()
      |      ^~~~~~~
y.tab.c:1405:7: note: previous implicit declaration of ‘yyerror’ was here
 1405 |       yyerror (YY_("syntax error"));
      |       ^~~~~~~
nirma@nirma-63:~/Downloads/calculator$ ./a.out

Enter Arithmatic Expression : 12+23-3

Result = 32
Valid Expression!

Enter Arithmatic Expression : 8*6/2

Result = 24
Valid Expression!

Enter Arithmatic Expression : (5+3)*(8/2)-(-3)

Result = 35
Valid Expression!

Enter Arithmatic Expression :
```

Conclusion:

In this practical, we learnt how we can use yacc (yet another compiler compiler) to make a syntax directed translation program with help of lex. We can also make any type of grammer and use yacc to parse it.