

ASSIGNMENT: 04

Name: Abhijeet Biswas
SRN: 201900400
Roll No: 05
Div: B

Question:

Design a YACC and corresponding LEX specification to compute the value of an expression. Consider arithmetic, trigonometric, $1/x$, \sqrt{x} , x^y etc. operators.

Input:

1. $5+30*2$
2. $-6-10$
3. $\cos(90)+3$

Code:

ass4.y –

```
% {  
  
    #include <stdio.h>  
  
    #include <math.h>  
  
% }  
  
  
%union      //to define possible symbol types  
{ double p; }  
%token<p>num  
%token SIN COS TAN LOG SQRT  
  
  
%left '+' '-' //lowest precedence  
%left '*' '/' //highest precedence
```

```
%nonassoc uminu //no associativity
%type<p>exp      //Sets the type for non-terminal
%%
```

```
/*for storing the answer */
```

```
ss: exp {printf(" Answer = %g\n",$1);}
```

```
/* for binary arithmetic operators */
```

```
exp:  exp+'exp' { $$=$1+$3; }
      |exp '-'exp { $$=$1-$3; }
      |exp '*'exp { $$=$1*$3; }
      |exp '/'exp {
          if($3==0)
          {
              printf("Divide by Zero");
          }
          else $$=$1/$3;
      }
      | '-'exp { $$=-$2;}
      |SIN('exp') { $$=sin($3);}
      |COS('exp') { $$=cos($3);}
      |TAN('exp') { $$=tan($3);}
      |LOG('exp') { $$ =log($3);}
      |SQRT('exp') { $$ =sqrt($3);}
      |num;
      |('exp') { $$=$2;}
```

%%

```
int main()
```

```
{
```

```
    do
```

```
    {
```

```
        printf("\nExpression:");
```

```
        yyparse();    /* Parse the sentence repeatedly until the i/p runs out
*/
```

```
    }while(1);
```

```
}
```

```
yyerror(char *s;) /* to print error message when an error is parsing of i/p */
```

```
{
```

```
    printf("Syntax Error");
```

```
}
```

ass4.1 –

```
% {
```

```
    #include <math.h>
```

```
    #include "y.tab.h"
```

```
% }
```

%%

```
[0-9]+|[0-9]*\.[0-9]+ {
```

```
    yylval.p = atof(yytext);
```

```
    return num;
```

}

sin {return SIN;}

cos {return COS;}

tan {return TAN;}

log {return LOG;}

sqrt {return SQRT;}

[\t] ;

\n return 0;

. return yytext[0];

%%

Output:

```
oblivion@oblivion-VirtualBox: ~/Documents/TY/Sem VI/CD/Ass4
oblivion@oblivion-VirtualBox:~/Documents/TY/Sem VI/CD/Ass4$ lex ass4.l
oblivion@oblivion-VirtualBox:~/Documents/TY/Sem VI/CD/Ass4$ yacc ass4.y
oblivion@oblivion-VirtualBox:~/Documents/TY/Sem VI/CD/Ass4$ yacc -d ass4.y
oblivion@oblivion-VirtualBox:~/Documents/TY/Sem VI/CD/Ass4$ gcc lex.yy.c y.tab.c -ll -lm
y.tab.c: In function 'yyparse':
y.tab.c:1259:16: warning: implicit declaration of function 'yylex' [-Wimplicit-function-declaration]
1259 |     yychar = yylex ();
      |                ^~~~~
y.tab.c:1464:7: warning: implicit declaration of function 'yyerror'; did you mean 'yyerrok'? [-Wimplicit-function-declaration]
1464 |     yyerror (YY_("syntax error"));
      |     ^~~~~~
      | yyerrok
ass4.y: At top level:
ass4.y:51:1: warning: return type defaults to 'int' [-Wimplicit-int]
51 | yyerror(char *s;) /* to print error message when an error is parsing of i/p */
    | ^~~~~~
oblivion@oblivion-VirtualBox:~/Documents/TY/Sem VI/CD/Ass4$ ./a.out

Expression:5+30*2
Answer = 65

Expression:-6-10
Answer = -16

Expression:cos(90)+3
Answer = 2.55193

Expression:
```