ESCUELA POLITÉCNICA NACIONAL
FACULTAD DE INGENIERÍA DE SISTEMAS CIENCIAS DE LA COMPUTACIÓN

COMPILADORES Y LENGUAJES

Informe – Prueba 6

GRUPO: GR1COM

FECHA DE ENTREGA: 12/03/2021

INTEGRANTES: Alejandro Moya, Leonardo Andrade, Jefferson Cando

---

Usamos LEX y YACC para implementar una pequeña calculadora que nos permita realizar las operaciones aritméticas con operadores binarios como suma, resta, multiplicación, división y potencia con paréntesis y operandos de tipo entero y real. La calculadora a más de calcular el valor expone la secuencia en notación postfija de manera descrita.

CÓDIGO LEX

```
%{
#include <stdio.h>
#include "y.tab.h"
FILE *salida;
%}
%option noyywrap
%option yylineno
DIGITO [0-9]
ID [A-Za-z][a-zA-Z0-9_]*
%%
{DIGITO}+("."{DIGITO}+)? {yylval.real=atof(yytext); return(TKN_NUM);}
"="	{return(TKN_ASIGN);}
";"	{return(TKN_PTOCOMA);}
"*"	{return(TKN_MULT);}
"/"	{return(TKN_DIV);}
```

```
"+"     {return(TKN_MAS);}
"-"     {return(TKN_MENOS);}
"^"     {return(TKN_POW);}
"("     {return(TKN_PARENTESISI);}
")"     {return(TKN_PARENTESISD);}
{ID}    {return(TKN_ID);}
%%
int main (int argc, char *argv[]){
        if((yyin=fopen("ingreso.txt", "rt"))==NULL)
        {
                printf("\nNo se puede abrir el archivo: %s\n", argv[1]);
        }else
        {
        if((salida=fopen("salida.txt", "w"))== NULL)
        {
                printf("\nNo se puede abrir el archivo: %s\n", argv[1]);
        }
        yyparse();
        }
        fclose(salida);
        fclose(yyin);
return 0;
}
```

CÓDIGO YACC

```
%{
#include <stdio.h>
#include <math.h>
extern int yylex(void);
extern char *yytext;
FILE *salida;
void yyerror (char *s);
%}
%union
{
        float real;
}
%start Calculadora
%token          <real> TKN_NUM
%token          TKN_ASIGN
%token          TKN_PTOCOMA
%token          TKN_MULT
%token          TKN_DIV
%token          TKN_MAS
%token          TKN_MENOS
%token          TKN_POW
%token          TKN_PARENTESISI
%token          TKN_PARENTESISD
```

```
%token          <real> TKN_ID
%type  <real> Expresion
%left TKN_MAS TKN_MENOS
%left TKN_MULT TKN_DIV
%right TKN_POW
%%
Calculadora: TKN_ID {
        printf("El resultado de %s con los siguientes TOKENS encontrados es : \nidentificador,
asigancion, ",yytext);
        fprintf(salida,"El resultado de %s con los siguientes TOKENS encontrados es : \
nidentificador, asigancion, ",yytext);
        }
TKN_ASIGN Expresion TKN_PTOCOMA
        {
        printf("punto y coma =%5.2f",$4);
        fprintf(salida,"punto y coma =%5.2f",$4);
        };
Expresion : TKN_NUM
        {
        $$=$1;
        printf ("numero positivo, ");
        fprintf (salida,"numero positivo, ");
        }|
    TKN_MENOS TKN_NUM
        {
        $$=-$2;
        printf("NUmero negativo, ");
        fprintf(salida,"NUmero negativo, ");
        }|
    TKN_PARENTESISI TKN_NUM TKN_PARENTESISD
        {
        $$=$2;
        printf("abre parentesis, numero positivo, cierra parentesis, ");
        fprintf(salida,"abre parentesis, numero positivo, cierra parentesis, ");
        }|
        TKN_PARENTESISI TKN_MENOS TKN_NUM TKN_PARENTESISD
        {
        $$=-$3;
        printf("abre parentesis, numero negativo, cierra parentesis, ");
        fprintf(salida,"abre parentesis, numero negativo, cierra parentesis, ");
        }|
        Expresion TKN_POW Expresion
        {
        $$=pow($1,$3);
        printf("operador exponencial, ");
        fprintf(salida,"operador exponencial, ");
        }|
        TKN_PARENTESISI Expresion TKN_POW Expresion TKN_PARENTESISD
        {
        $$=pow($2,$4);
        printf("abre parentesis, operador exponencial, cierra parentesis, ");
```

```
        fprintf(salida,"abre parentesis, operador exponencial, cierra parentesis, ");
        }|
        Expresion TKN_MAS Expresion
        {
        $$=$1+$3;
        printf("operador mas ");
        fprintf(salida,"operador mas ");
        }|
        TKN_PARENTESISI Expresion TKN_MAS Expresion TKN_PARENTESISD
        {
        $$=$2+$4;
        printf("abre parentesis, operador mas, cierra parentesis, ");
        fprintf(salida,"abre parentesis, operador mas, cierra parentesis, ");
        }|
        Expresion TKN_MENOS Expresion
        {
        $$=$1-$3;
        printf("operador menos ");
        fprintf(salida,"operador menos ");
        }|
        TKN_PARENTESISI Expresion TKN_MENOS Expresion TKN_PARENTESISD
        {
        $$=$2-$4;
        printf("abre parentesis, operador menos, cierra parentesis, ");
        fprintf(salida,"abre parentesis, operador menos, cierra parentesis, ");
        }|
        Expresion TKN_MULT Expresion
        {
        $$=$1*$3;
        printf("operador multiplicacion ");
        fprintf(salida,"operador multiplicacion ");
        }|
        TKN_PARENTESISI Expresion TKN_MULT Expresion TKN_PARENTESISD
        {
        $$=$2*$4;
        printf("abre parentesis, operador multiplicacion, cierra parentesis, ");
        fprintf(salida,"abre parentesis, operador multiplicacion, cierra parentesis, ");
        }|
        Expresion TKN_DIV Expresion
        {
        $$=$1/$3;
        printf("operador division ");
        fprintf(salida,"operador division ");
        }|
        TKN_PARENTESISI Expresion TKN_DIV Expresion TKN_PARENTESISD
        {
        $$=$2/$4;
        printf("abre parentesis, operador division, cierra parentesis, ");
        fprintf(salida,"abre parentesis, operador division, cierra parentesis, ");
        };
%%
```

```
void yyerror(char *s){
	printf("\nError %s", s);
	fprintf(salida,"\nError %s", s);

}
```

Entrada	Salida
X=1+2+3+4; X=2+2; X=2^2^2^2;	El resultado de X con los siguientes TOKENS encontrados es : identificador, asigancion, numero positivo, numero positivo, operador mas numero positivo, operador mas numero positivo, operador mas punto y coma =10.00 Error syntax error