

UNIVERSIDAD NACIONAL DE SAN AGUSTIN DE
AREQUIPA



Ciencia de la Computacion

COMPILADORES GRUPO A

YUBER ELMER VELAZCO PAREDES

PRACTICA 05

INTEGRANTES:

PUCHO ZEVALLOS KELVIN PAUL

2020

1. **Proponga una gramática para la evaluación de expresiones aritméticas. El programa debe leer un archivo de texto plano (programa1.txt), generar la forma postfija y mostrar el resultado.**

Programa1.txt

```
INICIO
    14 + 5;
    (12 - 6 ) + 8;
    ((8+12)-(5-2))*2;
FIN
```

1.1. Código Fuente

```
#include <stdio.h>
#include <ctype.h>
#include <iostream>
#include <string.h>

using namespace std;
#define MAS '+'
#define MENOS '-'
#define NUM 256
#define FIN -1
#define ID 257
#define INICIO 258
#define END 259
#define PARENTESISAPER '('
#define PARENTESISCIER ')'
#define PUNTOYCOMA ';'
#define MULTI '*'

char lexema[80];
int tok;

FILE* f;

void para(int);
void error();
void Resto();
void Term();
int scanner();

int cont=0;
int indice = 0;//indice para a pila
int result = 0;//las operacion son reservadas en result
int tam;
int Pila[80];
int resultados[80];//alamacena todas las operaciones que se realiza

int ind=0;//indice para los resultados
```

```

char guardar;
char nuevo = FIN;

void Exp()
{
    if (tok==NUM)
    {
        Term();
        Resto();
    }
    else if(tok == ID){
        if(tok+1 == INICIO && cont == 0){
            cout<<"INICIO\n\n\t";
            cont++;
        }
        else
        {
            para(ID);
            if( tok == EOF)
                cout<<"\nFIN\n";

        }
    }
    else{
        error();
    }
}

void Resto()
{
    if (tok == MAS)
    {
        para(MAS);

        if(tok == PARENTESISAPER){
            para(PARENTESISAPER);
        }
        Term();

        if(tok == MENOS || tok == MAS || tok == MULTI)
            ;
        else
            printf("+");

        if (tok == PUNTOYCOMA)
            cout<<"\n\t";
        if(tok == PARENTESISCIER)
            ;
        //operacion de pila
        result = 0;
        for(int i = 0; i<indice ; i++){
            result = result + Pila[i];
            Pila[i] = 0;
        }
        Pila[0] = result;
        indice = 1;
    }
}

```

```

        Resto();
    }
    else if (tok == MENOS)
    {

        para(MENOS);

        if(tok == PARENTESISAPER){
            nuevo = PARENTESISAPER;
            para(PARENTESISAPER);
        }
        Term();
        if(tok == MENOS || tok == MAS || tok == MULTI)
            ;
        else
            printf("-");

        if (tok == PUNTOYCOMA)
            cout<<"\n\t";
        if(tok == PARENTESISCIER)
            ;
        if(nuevo == PARENTESISAPER){
            result = Pila[1];
            Pila[1] = 0;
            for(int i = 2; i<indice ; i++){
                result = result - Pila[i];
                Pila[i] = 0;
            }
            Pila[1] = result;
            indice = 2;
        } else{

            //operacion de pila
            result = Pila[0];
            Pila[0] = 0;
            for(int i = 1; i<indice ; i++){
                result = result - Pila[i];
                Pila[i] = 0;
            }
            Pila[0] = result;
            indice = 1;
        }

        Resto();
    }
    else if(tok == MULTI){
        para(MULTI);
        Term();
        printf("*");
        if (tok == PUNTOYCOMA)
            cout<<"\n";
        //operacion de pila
        result = 1;
        for(int i = 0; i<indice ; i++){
            result = result * Pila[i];
            Pila[i] = 0;
        }
    }
}

```

```

    }
    Pila[0] = result;
    indice = 1;

    Resto();
}
else if (tok == PUNTOYCOMA)
{
    resultados[ind++]=result;
    indice = 0;
    Pila[0] = 0;
    result = 0;
    pareja(PUNTOYCOMA);
    if (tok == PARENTESISAPER){

        Resto();
    }
    else if (tok == NUM)
        Resto();
    else{

        Exp();
    }
} else if (tok == PARENTESISAPER){
    pareja(PARENTESISAPER);
    if ( tok == PARENTESISAPER){
        Resto();
    }
    else
        Exp();
} else if (tok == PARENTESISASCIER){
    nuevo = PARENTESISASCIER;
    pareja(PARENTESISASCIER);
    if (tok == MENOS || tok == MAS || tok == MULTI)
        guardar = tok;

    if ( tok == PARENTESISASCIER){
        cout<<guardar;

        result = Pila[0];
        for(int i = 1; i<indice ; i++){
            if(guardar == MAS)
                result = result + Pila[i];
            else if (guardar == MENOS)
                result = result - Pila[i];
            else
                result = result * Pila[i];

            Pila[i] = 0;
        }
        Pila[0] = result;
        indice = 1;

        pareja(PARENTESISASCIER);
        Resto();
    } else{

```

```

        Resto();
    }
}
else
    //cout <<"cadena vacia\n";
}
void Term()
{
    int term = 0;
    if (tok==NUM)
    {
        printf("%s",lexema);

        for(int j = 0 ; j <= tam ; j++){
            term = term + lexema[j] - '0';
            if((j+1) <= tam)
                term = term * 10;
        }
        Pila[indice++] = term;

        pareo(NUM);
    }
    else
        error();
}

void error()
{
    printf("Error de sintaxis");
}

void pareo(int t)
{
    if (tok==t){
        tok=scanner();
    }
    else{
        error();
    }
}

int scanner()
{
    int c,i;
    do c=fgetc(f);
    while(isspace(c));

    if(c == EOF) return EOF;

    if (c==MAS || c==MENOS || c==MULTI)
        return c;

    if(c == PARENTESISAPER || c == PARENTESISCIER || c == PUNTOYCOMA)
        return c;
}

```

```

    if (isalpha(c)){
        i=0;
        do
        {
            lexema[i++] = c;
            c = fgetc(f);
        } while (isalnum(c) || c == '_');

        lexema[i] = 0;
        ungetc(c,f);
        return ID;
    }
    if(isdigit(c))
    {
        i=0;
        do {
            lexema[i++] = c;
            c = fgetc(f);
        } while(isdigit(c));

        lexema[i] = 0;

        tam = i-1;

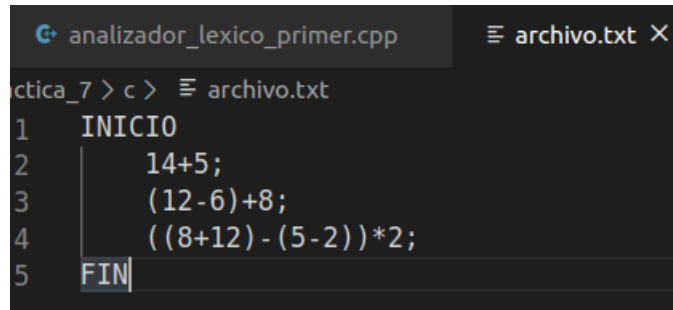
        ungetc(c,f);
        return NUM;
    }
}

int main()
{
    f = fopen("archivo.txt","r");
    while(1){
        tok=scanner();
        if(tok==EOF) break;
        Exp();
    }
    cout<<endl;
    for(int i=0;i<ind;i++)
        cout<<"operacion_"<<i+1<<"_="<<resultados[i]<<endl;

    cout<<endl;
    return 0;
}

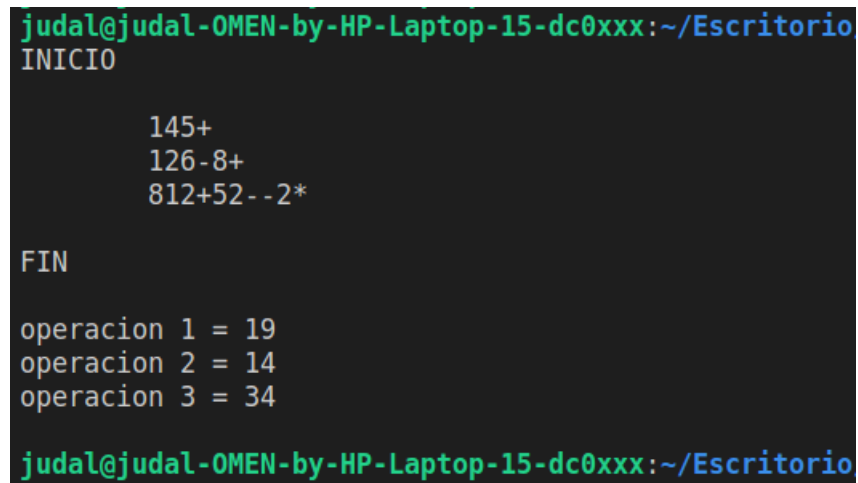
```

1.2. Captura del formato de texto plano



```
analizador_lexico_primer.cpp  archivo.txt X
ctica_7 > c >  archivo.txt
1  INICIO
2      14+5;
3      (12-6)+8;
4      ((8+12)-(5-2))*2;
5  FIN
```

1.3. Captura de pantalla del resultado



```
judal@judal-OMEN-by-HP-Laptop-15-dc0xxx:~/Escritorio
INICIO

      145+
      126-8+
      812+52--2*

FIN

operacion 1 = 19
operacion 2 = 14
operacion 3 = 34

judal@judal-OMEN-by-HP-Laptop-15-dc0xxx:~/Escritorio
```