

**Práctica Nro. 03**

Nombre: \_\_\_\_\_

**1.- Objetivo.**

Escribir analizadores léxicos.

**2.- Desarrollo.**

A) (5pts) Desarrollar un analizador léxico “scanner” que reconozca los siguientes tokens: NUM, ID, MAYOR, MAYORIGUAL, PUNTOYCOMA, PARI, WHILE, IF. La entrada de datos puede ser, o bien desde un archivo o desde la entrada estándar. Cada vez que se reconozca un token, imprima el nombre del token y el lexema.

Solución:

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

#define MAYOR    '>'
#define PUNTOYCOMA  '.'
#define PARI     '('
#define ID       256
#define NUM      257
#define MAYORIGUAL 258
#define WHILE    259
#define IF       260

int scanner();
void mostrar(int);
int espalres();

FILE *f;
char lexema[80];

void main(int n, char *pal[])
{
    int token;
    f=stdin; //entrada estandar del teclado
    if(n==2) //si se especifico un archivo de lectura
    {
        f=fopen(pal[1],"rt"); //lectura modo texto
        if(f==NULL)
            f=stdin;
    }
    if(f==stdin) //la lectura sera desde la entrada estandar
        printf("Ingrese texto ..... termine con Ctrl z \n");

    while(1)
    {
        token=scanner();
        if(token==EOF) break;
        mostrar(token);
    }

    if(f !=stdin) // si la entrada fue de un archivo
        fclose(f); // entonces cerrar el archivo.
}

//fin del main

int scanner()
{
    int c;
    int i;
    do c=fgetc(f); while(isspace(c)); //ignora blancos

    if(c==EOF) return EOF;

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

        lexema[i]=0;
        ungetc(c,f); //se devuelve c al flujo de entrada
        i=espalres(); // verifica si es palabra reservada
        // WHILE , IF

        if(i>=0)
            return i;
        return ID; // se trata de un ID
    }

    if(isdigit(c)) //regla del NUM
    {
        i=0;
        do{
            lexema[i++]=c;
            c=fgetc(f);
        }while(isdigit(c));

        lexema[i]=0;
        ungetc(c,f);
        return NUM;
    }

    //regla de PUNTOYCOMA y PARI
    if((c=='.' || c=='(')) return c; //regla del "." y "("

    if(c=='>') //regla de ">" o ">="
    {
        c=fgetc(f);
        if(c=='=') //return MAYORIGUAL
        {
            lexema[0]='>'; lexema[1]='='; lexema[2]=0;
            return MAYORIGUAL;
        }
        ungetc(c,f);
        return MAYOR; //return MAYOR
    }

}

//fin de scanner

int espalres()
{
    if(strcmp(lexema,"while")==0) return WHILE;
    if(strcmp(lexema,"if")==0) return IF;

    return -1;
}

void mostrar(int token)
{
    switch(token)
    {
        case ID: printf("ID: %s\n", lexema); break;
        case NUM: printf("NUM: %s\n", lexema); break;
        case MAYOR: printf("MAYOR: %s\n", lexema); break;
        case MAYORIGUAL: printf("MAYORIGUAL: %s\n", lexema); break;
        case PUNTOYCOMA: printf("PUNTOYCOMA: %s\n", lexema); break;
        case PARI: printf("PARI: %s\n", lexema); break;
        case WHILE: printf("WHILE: %s\n", lexema); break;
        case IF: printf("IF: %s\n", lexema); break;
    }
}
```

```
switch(token)
{
case ID: printf("token = ID [%s] \n",lexema); break;
case NUM: printf("token = NUM [%s] \n",lexema); break;
case MAYORIGUAL: printf("token = MAYORIGUAL [%s] \n",lexema); break;
case WHILE: printf("token = WHILE [%s] \n",lexema); break;
case IF: printf("token = IF [%s] \n",lexema); break;

case PARI: printf("token = PARI [%c] \n",token); break;
case MAYOR: printf("token = MAYOR [%c] \n",token); break;
case PUNTOYCOMA: printf("token = PUNTOYCOMA [%c] \n",token); break;
}
```

B) Escribir un scanner que reconozca los tokens que se menciona a continuación y realizar pruebas con varios tipos de archivos de entrada.

- ID
- NUM
- +, -, \*, /
- > , >= , < , <= , = , == , !=
- ( , ) , [ , ] , ' , ' , { , }
- Palabras reservadas
- Ignore comentario en línea y en bloque.

### 3.- Evaluación.

La evaluación consiste en:

- (3pts) Solución del ejercicio A en clase de sesión práctica. Deberá enviar la captura de pantalla del programa en ejecución.
- (17pts) Informe del ejercicio B resuelto en un documento en formato PDF. El informe debe contener: enunciado del problema, código fuente y capturas de pantalla del programa en ejecución. De darse el caso, también debe incluir los otros archivos de texto plano

### 4.- Tiempo de desarrollo.

Tiempo total en sesión práctica: 60 min.