­

**Juan Cruz Neira**

**Natalia Lembo**

**Javier Capristo**

**Compilador en Lenguaje C**

**Sintaxis y Semántica de los Lenguajes**

**Grupo 2 - Integrantes**

**Trabajo PRACTICO N°4**

## Objetivo

Compilar el código brindado por la catedra y analizar los errores que produce.

## Código

Al código brindado por la cátedra le agregamos algunos printf a modo de descriptivo de lo que está haciendo el programa en ese momento.

/\* Compilador del Lenguaje Micro (Fischer) \*/

#include <stdio.h>

#include <string.h>

#include <ctype.h>

#define NUMESTADOS 15

#define NUMCOLS 13

#define TAMLEX 32+1

#define TAMNOM 20+1

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Declaraciones Globales\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

FILE \* in;

typedef enum {

INICIO, FIN, LEER, ESCRIBIR, ID, CONSTANTE, PARENIZQUIERDO, PARENDERECHO, PUNTOYCOMA, COMA, ASIGNACION, SUMA, RESTA, FDT, ERRORLEXICO

} TOKEN;

typedef struct {

 char identifi[TAMLEX];

 TOKEN t; /\* t=0, 1, 2, 3 Palabra Reservada, t=ID=4 Identificador (ver enum) \*/

 } RegTS;

RegTS TS[1000] = { {"inicio", INICIO}, {"fin", FIN}, {"leer", LEER}, {"escribir", ESCRIBIR}, {"$", 99} };

typedef struct{

 TOKEN clase;

 char nombre[TAMLEX];

 int valor;

 } REG\_EXPRESION;

char buffer[TAMLEX];

TOKEN tokenActual;

int flagToken = 0;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Prototipos de Funciones\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

TOKEN scanner(); // el scanner

int columna(int c);

int estadoFinal(int e);

void Objetivo(void); // del PAS

void Programa(void);

void ListaSentencias(void);

void Sentencia(void);

void ListaIdentificadores(void);

void Identificador(REG\_EXPRESION \* presul);

void ListaExpresiones(void);

void Expresion(REG\_EXPRESION \* presul);

void Primaria(REG\_EXPRESION \* presul);

void OperadorAditivo(char \* presul);

REG\_EXPRESION ProcesarCte(void);

REG\_EXPRESION ProcesarId(void);

char \* ProcesarOp(void);

void Leer(REG\_EXPRESION in);

void Escribir(REG\_EXPRESION out);

REG\_EXPRESION GenInfijo(REG\_EXPRESION e1, char \* op, REG\_EXPRESION e2);

void Match(TOKEN t);

TOKEN ProximoToken();

void ErrorLexico();

void ErrorSintactico();

void Generar(char \* co, char \* a, char \* b, char \* c);

char \* Extraer(REG\_EXPRESION \* preg);

int Buscar(char \* id, RegTS \* TS, TOKEN \* t);

void Colocar(char \* id, RegTS \* TS);

void Chequear(char \* s);

void Comenzar(void);

void Terminar(void);

void Asignar(REG\_EXPRESION izq, REG\_EXPRESION der);

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Programa Principal\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

int main(int argc, char \* argv[])

{

    TOKEN tok;

    char nomArchi[TAMNOM];

    int l;

    /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Se abre el Archivo Fuente\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

    // verifica errores posibles

    if ( argc == 1 ) {

        printf("Verificando si se coloco el nombre del archivo fuente\n");

        printf("Debe ingresar el nombre del archivo fuente (en lenguaje Micro) en la linea de comandos\n"); return -1; }// no puso nombre de archivo fuente

    if ( argc != 2 ) {

        printf("Verificando la cantidad de argumentos\n");

        printf("Numero incorrecto de argumentos\n"); return -1; }//los argumentos deben ser 2

    strcpy(nomArchi, argv[1]);

    l = strlen(nomArchi);

    if ( l > TAMNOM ) {

        printf("Verificando que el nombre del archivo sea correcto\n");

        printf("Nombre incorrecto del Archivo Fuente\n"); return -1; }

    // requiere para compilar un archivo de extensión.m archivo.m

    if ( nomArchi[l-1] != 'm' || nomArchi[l-2] != '.' ) {

        printf("Verificando que la extension del archivo sea la correcta\n");

        printf("Nombre incorrecto del Archivo Fuente. Se espera extension .m\n"); return -1; }

    if ( (in = fopen(nomArchi, "r") ) == NULL ) {

        printf("Verificando lectura del archivo\n");

        printf("No se pudo abrir archivo fuente\n"); return -1;//no pudo abrir archivo

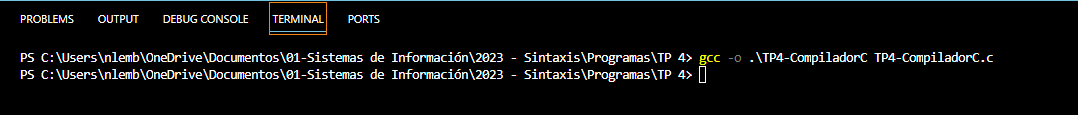
    }

    else

        printf("Soy un programa que funciona HERMOSO =)\n");

}

## Compilación del código:



## Programa en lenguaje micro correcto:

### Nombre del archivo: archivoMicroOk.m

inicio

    leer A;

    B := A - 45 + B;

    escribir (B);

fin;

## Programa en lenguaje micro con error léxico:

### Nombre del archivo: archivoMicroErrLexico.m

inicio

    leer A;

    B := A - 45 + B;

    escribir (B);

    BOOL B; //error léxico: La palabra clave BOOL no es reconocida

fin;

## Programa en lenguaje micro con error sintáctico y semántico:

### Nombre del archivo: archivoMicroErrSint.m

inicio

    leeer A; // Error sintáctico

    B := A - "45" + B; // Error semántico

    escribir (B);

fin;

## Ejecución del código pasando como argumento el programa correcto:

Captura de pantalla de computadora

Descripción generada automáticamente con confianza media

## Análisis de errores:

#### Verificación argc ==1

En esta parte del código se verifica que se pase el archivo fuente como argumento

Ejemplo: No se brinda ningún archivo

Una captura de pantalla de un celular con texto e imagen

Descripción generada automáticamente con confianza media

#### Verificación argc !=2

En esta parte del código se verifica que la cantidad de argumentos no sea distinta a dos.

Ejemplo: Se pasaron dos archivos, un total de 3 argumentos, por lo que envió el error

#### Verificación de la longitud del nombre del archivo

En esta parte del código se verifica que el nombre del archivo no sea mayor a TAMNOM = 21



#### Error en la extensión del archivo fuente

En esta parte del código se verifica que el archivo que se pase como argumento tenga la extensión “.m”. De no tenerla se envía el error y se notifica cual es la extensión esperada.

Ejemplo: Se pasa como argumento un archivo sin extensión.



#### Error de lectura

En esta parte del código se verifica que se pueda abrir el archivo con permiso de lectura (“r”:read). Puede ser que el archivo no tenga los permisos o que se escriba mal el nombre y por eso no se encuentre el archivo.

#### Ejemplo: Se compilo colocando una “o” menos en el nombre y como no lo encontró no lo pudo abrir.

