

Avances en la Construcción de tu Traductor II

Objetivo:

Desarrollar un componente específico de un traductor (puede ser un compilador o un intérprete) para un lenguaje de programación simplificado. Este proyecto se realizará en varias fases, cada una centrada en una parte diferente del proceso de traducción.

Fase Actual: Análisis Léxico y sintactico

Descripción:

En esta fase, deberás implementar el analizador léxico para el lenguaje de programación asignado. El lenguaje tendrá una sintaxis y un conjunto de tokens definidos previamente por el instructor. El analizador léxico deberá ser capaz de leer el código fuente y convertirlo en una secuencia de tokens que serán utilizados en fases posteriores del proceso de traducción.

Requerimientos: Basado en la especificación del lenguaje proporcionada, define los tokens que formarán parte del lenguaje. Esto incluye palabras reservadas, identificadores, literales numéricos, operadores, etc.

: Escribe tu propio analizador en el lenguaje de programación de tu elección. El analizador debe poder leer un archivo de entrada con código fuente y producir una lista de tokens.

: Tu analizador léxico debe ser capaz de manejar y reportar errores léxicos de manera adecuada, como caracteres inválidos o formatos incorrectos de tokens.

Entregables: El código fuente de tu analizador léxico y sintactico, incluyendo cualquier estructura de datos utilizada para almacenar los tokens.

: Un breve documento que explique tu diseño, las decisiones importantes que tomaste durante la implementación, y cómo se manejan los errores léxicos y sintacticos.

Capturas de pantalla de la ejecución:

```
-----
4 | float | tipo
-----
0 | hola | identificador
-----
13 | , | ,
-----
0 | adios | identificador
-----
12 | ; | ;
-----
4 | int | tipo
-----
0 | c | identificador
-----
12 | ; | ;
-----
4 | void | tipo
-----
0 | funcion | identificador
-----
14 | ( | (
-----
4 | int | tipo
-----
0 | uno | identificador
-----
13 | , | ,
-----
4 | float | tipo
```

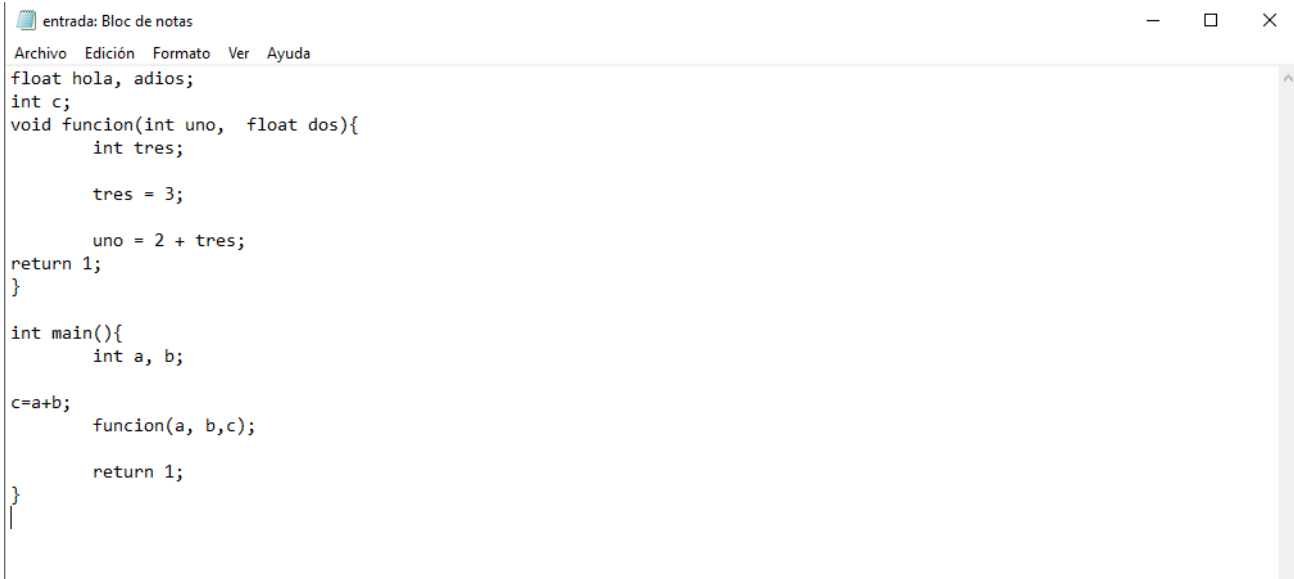
```
-----
0 | dos | identificador
-----
15 | ) | )
-----
16 | { | {
-----
4 | int | tipo
-----
0 | tres | identificador
-----
12 | ; | ;
-----
0 | tres | identificador
-----
18 | = | =
-----
1 | 3 | entero
-----
12 | ; | ;
-----
0 | uno | identificador
-----
18 | = | =
-----
1 | 2 | entero
-----
5 | + | opSuma
-----
0 | tres | identificador
```

MUESTRA DE LOS ERRORES:

```
21 | return | return
-----
1 | 1 | entero
-----
12 | ; | ;
-----
17 | } | }
-----
23 | $ | $

aceptacion
ERROR 6: "" El Valor Regresa NO es del mismo tipo que la funcion
-----
hola | global | f
-----
adios | global | f
-----
c | global | i
-----
uno | funcion | i
-----
dos | funcion | i
-----
Funcion | global | v
-----
tres | funcion | i
```

ARCHIVO DE ENTRADA:



```
entrada: Bloc de notas
Archivo Edición Formato Ver Ayuda
float hola, adios;
int c;
void funcion(int uno, float dos){
    int tres;

    tres = 3;

    uno = 2 + tres;
return 1;
}

int main(){
    int a, b;

c=a+b;
    funcion(a, b,c);

    return 1;
}
```

COMPLICACIONES:

La verdad si se me complico la elaboración de este si cumplio unas características me tarde bastante rato en elaborarlo y finalmente creo que se cumplio aunque no esta del todo organizado pero es entendible le agregue varias cosas para que se pudiera entender y elaborar mejor.