# Tokens

1. Utiliza extern char strToken para almacenar los nombres de cada tipo de token.
2. Define una enumeración con los tipos de orden llamada tipoToken.
3. Estructura TToken:

* tipoToken tipo: como su nombre lo indica, este campo guarda el tipo de token que es, siendo los tipos:

1. operador
2. numero
3. variable
4. paréntesis
5. igual

* char token [20]: esta cadena de caracteres es en donde de hecho se guarda el dato de cada token.

1. Int esOperador (char c)

* *Función:* Verifica si el carácter pasado como parámetro es un operador indicado.
* *Pasos:*

1. Crea un entero de nombre k.
2. Crea un arreglo static char llamado operadores, que contiene los operadores +, -, \*, / y ^.
3. Entra un ciclo for con la variable k como contador, desde 0 hasta el tamaño de la cadena de caracteres operadores.
4. Dentro del ciclo, entra una condición que verifica si el parámetro c es igual a algún operador. Si la condición se cumple, la función regresa un uno.
5. Regresa un cero.
6. Int esParentesis (char c)

* *Función:* Verifica si el carácter pasado como parámetro es un paréntesis.
* *Pasos:*

1. Entra una condición. Si c es igual a (, ), [ o ] la función regresa un uno.
2. Regresa un cero.
3. TToken token (char \*s)

* *Función:* Crea un solo token con la cadena recibida como parámetro, identificando su tipo.
* *Pasos:*

1. Crea un arreglo static char llamado cadena con mil espacios.
2. Crea un apuntador static char llamado actual.
3. Crea un entero k.
4. Crea una cadena de caracteres con 100 espacios llamada res.
5. Crea un apuntador de tipo TToken llamado unToken.
6. Entra una condición. Si la dirección s pasada como parámetro es diferente a null, la cadena en s se copia a cadena y la dirección de cadena se guarda en actual.
7. Entra un ciclo while que se ejecuta mientras el contenido de actual sea un espacio, un salto de línea o un tabulador. Dentro de este ciclo, se aumenta una unidad a actual con aritmética de apuntadores.
8. Entra una condición. Si el contenido de actual es un carácter nulo, se le asigna una dirección de memoria a unToken a través de la función malloc con tamaño de TToken, se copia el texto “Termine” al campo token de unToken y regresa el contenido de unToken.
9. Entra la primera condición de identificación de token, en la cual se usa la función esOperador para verificar si el contenido de actual es un operador. Si el contenido de actual es un operador, copia el operador a res con índice cero, se le asigna un carácter nulo a res con índice uno, se le asigna una dirección de memoria a unToken, se suma una unidad a actual, copia res a el campo token de unToken, asigna el tipo operador al campo tipo de unToken y regresa el contenido de unToken.
10. Entra la segunda condición de identificación de token, en la cual se compara el contenido de actual con el símbolo igual. Se esta condición se cumple, realiza las mismas instrucciones de la primera condición y asigna el tipo igual al campo tipo de unToken.
11. Entra la tercera condición de identificación de token, en la cual se utiliza la función esParentesis para verificar si el contenido de actual es un paréntesis. Si la condición se cumple, realiza las mismas instrucciones de la primera y segunda condición y asigna el tipo paréntesis al campo tipo de unToken.
12. Entra la cuarta condición de identificación de token, en la cual se utiliza la función isdigit para verificar si el contenido de actual es un número o si el contenido de actual es un punto. Si la condición se cumple, asigna el valor cero a k y entra un ciclo while que se ejecuta mientras el contenido de actual no sea un carácter nulo y sea un número o un punto. Dentro del ciclo se guarda el contenido actual en res con índice k, se aumenta una unidad a actual y se aumenta una unidad a k.  
    Después del ciclo, asigna un carácter nulo a res con índice k, asigna una dirección de memoria con tamaño TToken a unToken, se copia res a el campo token de unToken, asigna el tipo numero al campo tipo de unToken y regresa el contenido de unToken.
13. Entra la quinta condición de identificación de token, en la cual se utiliza la función isalpha para verificar si el contenido de actual es una letra. Si la condición se cumple, realiza las mismas instrucciones que la cuarta condición, pero la condición del ciclo while es modificada para verificar que el contenido de actual no sea carácter nulo y sea una letra y asigna el tipo variable al campo tipo de unToken.
14. Si no se cumple ninguna condición, imprime los textos “Error: expresión incorrecta” y “Programa detenido”.
15. Se cierra el programa.