Manual Técnico Proyecto 02 - Compiladores 01

Nativas
Run

Especificaciones De La Aplicación

Framework Front End : Angular CLI version 13.3.3

Framework Back End : Node 16.13.1

IDE Utilizado: Visual Studio Code

Sistemas Operativos Compatibles : Windows 8 o superiores

Memoria Ram Recomendada : 4GB como mínimo.

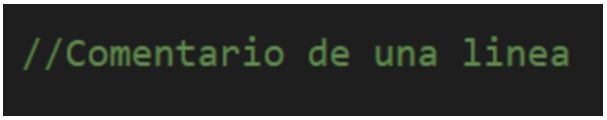
Descripción Del Lenguaje

El lenguaje no distinguirá entre mayúsculas o minúsculas.

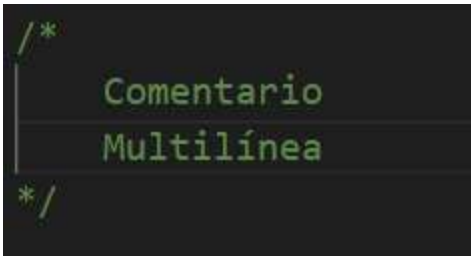
Comentarios

Los comentarios son una forma elegante de indicar que función tiene cierta sección del código que se ha escrito simplemente para dejar algún mensaje en específico. El lenguaje deberá soportar dos tipos de comentarios que son los siguientes:

Comentario de una línea:



Comentario Multilínea:



Tipos De Datos

```
int numero=90;  //Tipo de Dato Int
string letra="a"; //Tipo de Dato String
char letraChar='E'; //Tipo de Dato Char
boolean booleano=true; //Tipo de Dato Boolean
double numeroDouble=10.25; //Tipo de Dato Double
```

Operadores Aritméticos

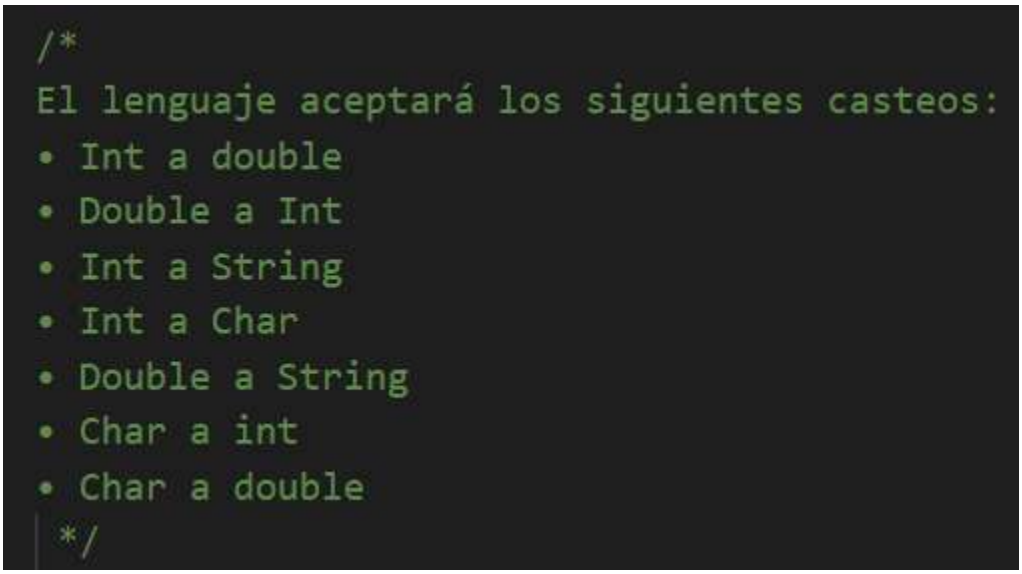
```
/*
+  Suma
-  Resta
* Multiplicación
/ Division
^ Potencia
% Módulo
*/
```

Operadores Relacionales

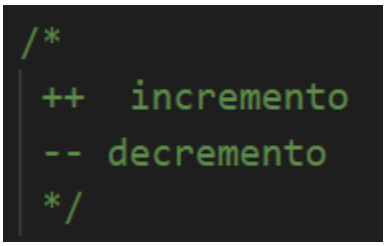
```
/*
== igualación
!= no igualación
< menor que
> mayor que
<= menor o igual que
>= mayor o igual que
*/
```

Operadores Lógicos

Casteos



Incremento Y Decremento



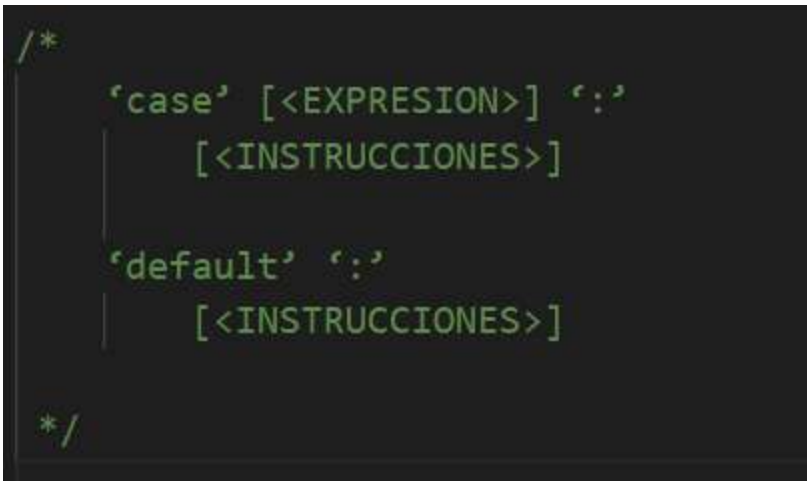
Sentencias De Control

```
/*

Existen : If, Else, IF Else
        'if' '(' [<EXPRESION>] ')' '{'
        [<INSTRUCCIONES>]
        '}'
        | 'if' '(' [<EXPRESION>] ')' '{'
        [<INSTRUCCIONES>]
        '}' 'else' '{'
        [<INSTRUCCIONES>]
        '}'
        | 'if' '(' [<EXPRESION>] ')' '{'
        [<INSTRUCCIONES>]
        '}' 'else' [<IF>]

*/
```

Switch

```
/*
    'switch' '(' [<EXPRESION>] ')' '{'
    [<CASES_LIST>] [<DEFAULT>]
    '}'
    | 'switch' '('<EXPRESION> ')' '{'
    [<CASES_LIST>]
    '}'
    | 'switch' '('<EXPRESION> ')' '{'
    [<DEFAULT>]
    '}'

*/
```

Case



While

For

Do-While

```
/*
    'do' '{'
        [<INSTRUCCIONES>]
    '}' 'while' '('[<EXPRESION>] ')' ';'

 */
```

Funciones

```
/*
    <ID> '(' [<PARAMETROS>] ')' ':' <TIPO> '{'
        [<INSTRUCCIONES>]
    '}'
    PARAMETROS -> [<PARAMETROS] ',' [<TIPO>] [<ID>]
                | [<TIPO>] [<ID>]
 */
```

Métodos

```
/*
    <ID> '(' [<PARAMETROS>] ')' ':' ['void'] '{'
        [<INSTRUCCIONES>]
    '}'
    PARAMETROS -> [<PARAMETROS] ',' [<TIPO>] [<ID>]
                | [<TIPO>] [<ID>]

 */
```

Print And Println

```
Print: No hay salto de Linea:
    'Print' '(' <EXPRESION> ')';
Println: Si hay salto de línea:
    'Println' '(' <EXPRESION> ')';
```

Nativas

```
'toLower' '(' <EXPRESION> ')';  ->Todo a minúscula.
'toUper' '(' <EXPRESION> ')';  ->Todo a mayúscula.
'Round' '(' <EXPRESION> ')';  ->Redondea un número.
'typeOf' '(' <EXPRESION> ')';  ->Devuelve el tipo de dato de la expresión.
'length' '(' <EXPRESION> ')';  ->Longitud de arreglo o cadena.
```

Run

Para poder ejecutar todo el código generado dentro del lenguaje, se utilizará la sentencia RUN
para indicar que método o función es la que iniciará con la lógica del programa.

```
'run <ID> '(' ')' ';'
'run' <ID> '(' <LISTAVALORES> ')' ';'
LISTAVALORES->LISTAVALORES ',' EXPRESION
            | EXPRESION
```