



Universidad de San Carlos de Guatemala  
Facultad de Ingeniería  
Escuela de Ingeniería en ciencias de sistemas.

## **Manual de usuario**

### **Nombre del estudiante**

José Luis Espinoza Jolón - 202202182

**Ing.(a).** Kevin Lajpop

Guatemala, abril 21 de 2024

## Contenido

|                                       |                               |
|---------------------------------------|-------------------------------|
| • Descripción .....                   | 3                             |
| • Entorno de trabajo .....            | 4                             |
| 1. Crear archivo (+): .....           | 5                             |
| 2. Abrir Archivo .....                | 5                             |
| Otras funcionalidades: .....          | 6                             |
| ▪ Guardar .....                       | 6                             |
| 3. Ejecutar .....                     | 6                             |
| 4. Reportes .....                     | 7                             |
| ▪ Reporte de errores .....            | 7                             |
| ▪ Reporte de Tabla de Símbolos: ..... | 7                             |
| ▪ Árbol Ast: .....                    | 7                             |
| 5. Area de entrada .....              | 7                             |
| 6. Area de consola: .....             | 7                             |
| • Uso del lenguaje .....              | 7                             |
| Parámetros iniciales: .....           | 7                             |
| Ejemplos de los reportes: .....       | 12                            |
| ▪ Token .....                         | ¡Error! Marcador no definido. |
| ▪ Errores .....                       | ¡Error! Marcador no definido. |
| ▪ Símbolos .....                      | ¡Error! Marcador no definido. |

## Descripción

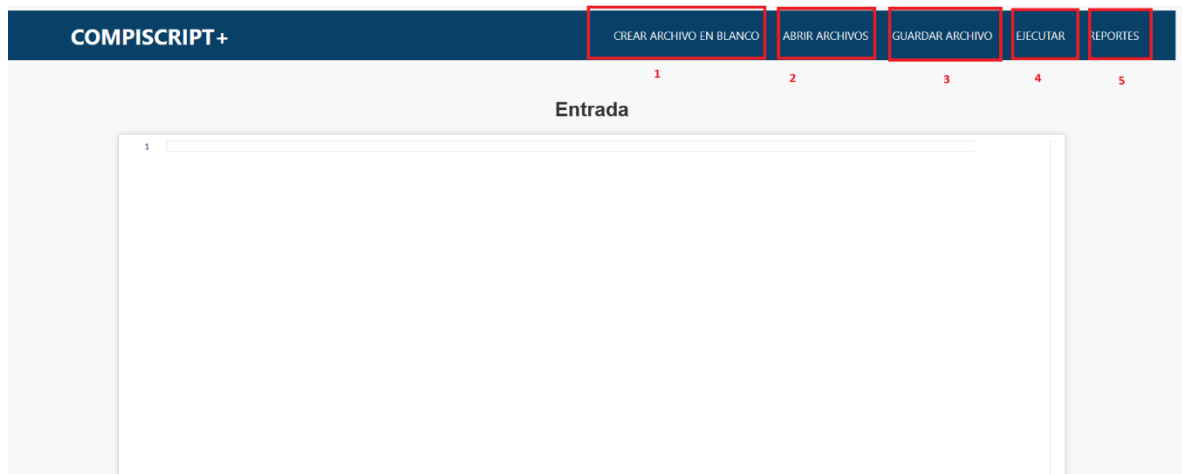
El curso de Organización de Lenguajes y Compiladores 1, ha puesto en marcha un nuevo proyecto, requerido por la Escuela de Ciencias y Sistemas de la Facultad de Ingeniería, que consiste en crear un lenguaje de programación para que los estudiantes del curso de Introducción a la Programación y Computación 1 aprendan a programar y tener conocimiento de todas las generalidades de un lenguaje de programación. Cabe destacar que este lenguaje será utilizado para generar sus primeras prácticas de

laboratorio del curso antes mencionado.

Por tanto, a usted, que es estudiante del curso de Compiladores 1, se le encomienda realizar el proyecto llamado CompiScript+. dado sus altos conocimientos en temas de análisis léxico, sintáctico y semántico

## **Entorno de trabajo**

El usuario al ejecutar el programa visualizara la siguiente vista.



Las siguientes funciones que puede usar el usuario es el siguiente.

1. **Crear archivo (+):** El editor debe tener la capacidad de crear archivos en blanco el cual podrá ser editado en una pestaña que tiene el nombre del archivo, se puede crear múltiples.  
De la siguiente manera se visualizará una nueva pestaña en la vista.



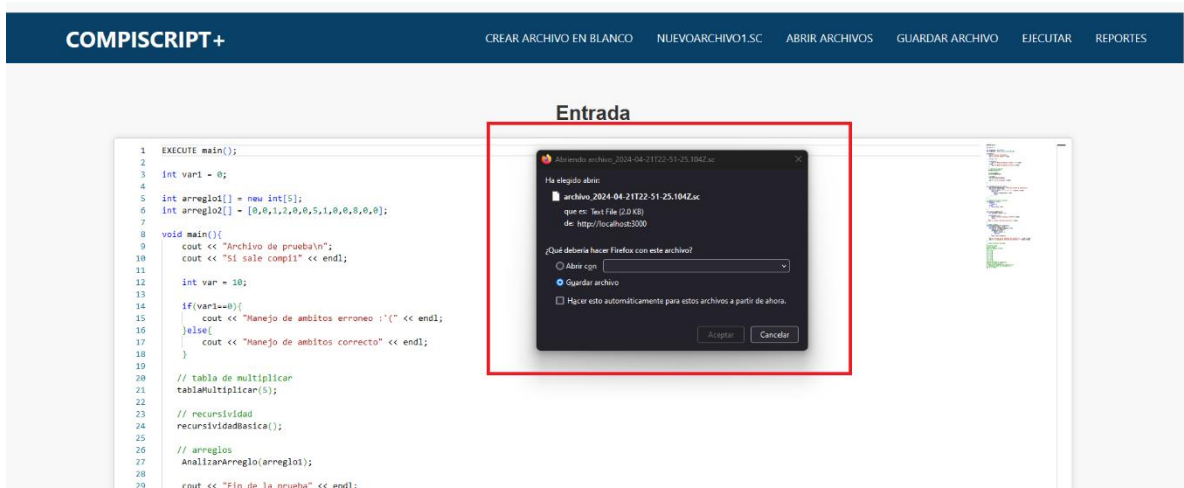
2. **Abrir Archivo:** El editor debe tener la capacidad de abrir archivos con las extensiones .sc cuyo contenido se deberá mostrar en el área de entrada en una nueva pestaña con el nombre del archivo.



De la siguiente manera se visualizará un archivo cargado.

### Otras funcionalidades:

- **Guardar:** El editor debe tener la capacidad de guardar el estado del archivo en el que se estará trabajando.



3. **Ejecutar:** se envía la entrada de la pestaña actualmente seleccionada

al intérprete con la finalidad de realizar el análisis, léxico, sintáctico y la ejecución de instrucciones.


#### 4. Reportes: Existe 3 tipos de reportes

- **Reporte de errores:** se mostrarán todos los errores léxicos y sintácticos encontrados.
- **Reporte de Tabla de Símbolos:** se mostrarán todas las variables y arreglos declarados.
- **Árbol Ast:** Se muestra todas las instrucciones

#### 5. Area de entrada: Se va a visualizar el código generado por el usuario.

#### 6. Area de consola:

En la consola de salida se mostrarán los resultados, mensajes y todo lo que sea indiciado en el lenguaje. Tiene como restricción el no ser editable por el usuario y únicamente puede mostrar información.



```
1 Archivo de prueba
2
3 Si sale compii
4 Manejo de ambitos correcto
5 5 x 1 = 5
6 5 x 2 = 10
7 5 x 3 = 15
8 5 x 4 = 20
9 5 x 5 = 25
10 5 x 6 = 30
11 5 x 7 = 35
12 5 x 8 = 40
13 5 x 9 = 45
14 5 x 10 = 50
15 5 x 11 = 55
16 Final de la tabla de multiplicar
17 Funcion recursiva correcta
18 Fin de la prueba
19
```

## Uso del lenguaje

### Parámetros iniciales:

Todas las sentencias del lenguaje deben venir entre EXECUTE

## **Comentarios:**

Existe 2 tipos de Comentarios, los comentarios les pueden ayudar al usuario a la hora de describir una parte de código o para que sirva y recordarse

**// Esto es un comentario de una sola línea**

**/\* Esto es un comentario**

**Multilínea \*/**

## **Tipos de Dato**

Los tipos de dato que soportará el lenguaje en concepto de un tipo de variable se definen a continuación:

tipos: Entero, double, bool, char, cadena

## **Operadores Aritméticos**

Suma, Es la operación aritmética que consiste en realizar la suma entre dos o más valores. Para esta se utiliza el signo más (+).

Resta, Es la operación aritmética que consiste en realizar la resta entre dos o más valores. Para esta se utiliza el signo menos (-).

Multiplicación, Es la operación aritmética que consiste en sumar un número (multiplicando) tantas veces como indica otro número (multiplicador). El signo para representar la operación es el asterisco (\*).

Division, Es la operación aritmética que consiste en partir un todo en varias partes, al todo se le conoce como dividendo, al total de partes se le llama divisor y el resultado recibe el nombre de cociente. El operador de la división es la diagonal (/).

Potencia, Es una operación aritmética de la forma  $\text{pow}(a,b)$  donde  $a$  es el valor de la base y  $b$  es el valor del exponente que nos indicará cuantas veces queremos multiplicar el mismo número. Por ejemplo  $\text{pow}(5, 3)$ ,  $a=5$  y  $b=3$  tendríamos que multiplicar 3 veces 5 para obtener el resultado



final; 5x5x5 que da como resultado 125.

### **Casteos**

El lenguaje aceptará los siguientes casteos:

- int a double
- double a int
- int a string
- int a char
- double a string
- char a int
- char a double

### **Sentencias de control**

Estas sentencias modifican el flujo del programa introduciendo condicionales. Las sentencias de control para el programa son el IF y el SWITCH.

### **Sentencias cíclicas**

Los ciclos o bucles son una secuencia de instrucciones de código que se ejecutan una vez tras otra mientras la condición, que se ha asignado para que pueda ejecutarse, sea verdadera. En el lenguaje actual, se podrán realizar 3 sentencias cíclicas que se describen a continuación.

### **Funciones**

Una función es una subrutina de código que se identifica con un nombre, tipo y un conjunto de parámetros. Para este lenguaje las funciones serán declaradas definiendo primero su tipo, luego un identificador único, seguido de una lista de parámetros dentro de paréntesis (esta lista de parámetros puede estar vacía en el caso de que la función no utilice parámetros).

Cada parámetro debe estar compuesto por su tipo seguido de un identificador, para el caso de que sean varios parámetros se debe utilizar

comas para separar cada parámetro y en el caso de que no se usen parámetros no se deberá incluir nada dentro de los paréntesis. Luego de definir la función y sus parámetros se declara el cuerpo de la función, el cual es un conjunto de instrucciones delimitadas por llaves {}.

Para las funciones es obligatorio que las mismas posean un valor de retorno que coincida con el tipo con el que se declaró la función, en caso de que no sea el mismo tipo o de que no venga un retorno dentro del cuerpo de la función debería lanzarse un error de tipo semántico.

## **Métodos**

Un método también es una subrutina de código que se identifica con un tipo, nombre y un conjunto de parámetros, aunque a diferencia de las funciones estas subrutinas no deben de retornar un valor. Para este lenguaje los métodos serán declarados haciendo uso de la palabra reservada 'void', seguido de un identificador del método, seguido de una lista de parámetros dentro de paréntesis (esta lista de parámetros puede estar vacía en el caso de que la función no utilice parámetros).

Cada parámetro debe estar compuesto por su tipo seguido de un identificador, para el caso de que sean varios parámetros se debe utilizar comas para separar cada parámetro y en el caso de que no se usen parámetros no se deberá incluir nada dentro de los paréntesis. Luego de definir el método y sus parámetros se declara el cuerpo del método, el cual es un conjunto de instrucciones delimitadas por llaves {}.

## **Llamada**

La llamada a una función específica la relación entre los parámetros reales y los formales y ejecuta la función. Los parámetros se asocian normalmente por posición, aunque, opcionalmente, también se pueden asociar por nombre. Si la función tiene parámetros formales por omisión, no es necesario asociarles

un parámetro real.

La llamada a una función devuelve un resultado que ha de ser recogido, bien asignándole a una variable del tipo adecuado, bien integrándose en una expresión.

### Salida en consola , con errores

```

1  ---> lexico:token no reconocido: # Fila: 7 Columna: 0
2  ---> lexico:token no reconocido: # Fila: 53 Columna: 0
3  Archivo de prueba
4
5  Si sale compil
6  Manejo de ambitos correcto
7  5 x 1 = 5
8  5 x 2 = 10
9  5 x 3 = 15
10 5 x 4 = 20
11 5 x 5 = 25
12 5 x 6 = 30
13 5 x 7 = 35
14 5 x 8 = 40
15 5 x 9 = 45
16 5 x 10 = 50
17 5 x 11 = 55
18 Final de la tabla de multiplicar
19 Funcion recursiva correcta
20 Fin de la prueba
21
```

## Ejemplos de los reportes:

- AST

