



# USER

## MANUAL

Guía de utilización VLANG-CHERRY

### Prepared By

KEVIN ANDRES ALVARES HERRERA - 202203038

MARCO POOL CHICHÉ SAPÓN - 3357975470901

VICENTE ROCAEL MATÍAS OSORIO - 3208209201308

Símbolos

Descripción

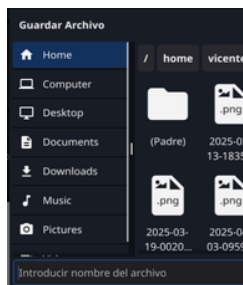
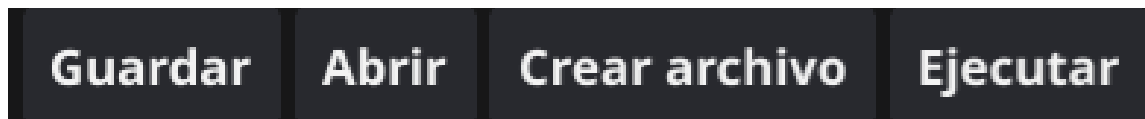
# 1. ELEMENTOS DEL SISTEMA

## 1.1 Fragmentos del Sistema

Hay 5 partes esenciales dentro del sistema

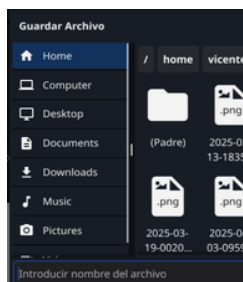
1. **Menu:** Guardar, Abrir, Crear Archivo, Ejecutar
2. **Editor:** Ámbito para escribir código vlang, editar eliminar, código en tiempo real.
3. **Consola:** Apartado en donde se visualiza la salida del análisis del fragmento de entrada plasmado en el editor después de presionar el botón Ejecutar
4. **Tabla de Símbolos/Errores:** En éste apartado se visualiza la información de los fragmentos analizados.
5. **Árbol CST:** Después del análisis, si no tiene errores se muestra el árbol cst

## 1.2 Menú Bar



### Botón Guardar

Se abre un submenú y éste permite guardar el archivo que se está editando o creando en el editor



### Botón Abrir

Se abre un submenú y éste permite abrir un archivo .vch, el editor automáticamente se llena de la información que contiene dicho archivo

```
Guardar Abrir Crear archivo Ejecutar

// Funciones no recursivas
fn saludar() {
    println("¡Hola, mundo!")
}
fn saludar_persona(n string) {
    println("¡Hola, mundo!", n)
}

fn obtener_numero() int {
    return 42
}

fn sumar(a int, b int) int {
    return a + b
}

// Funciones recursivas
/*
fn factorial(n int) int {
    if n <= 1 {
        return 1
    }
}

Análisis completado
=== Archivo de prueba de funciones ===
===== Funciones no recursivas sin parámetros =====
¡Hola, mundo!
El número obtenido es: 42
OK Funciones sin parámetros: correcto

===== Funciones no recursivas con parámetros =====
¡Hola, mundo! Juan
La suma de 10 y 20 es: 30
OK Función con parámetros: correcto

===== strconv.Atoi =====
123 convertido a int: 123
```

## Botón Ejecutar

Cuando el editor contiene información y se presiona el botón, empieza el análisis de dicha información

## 2. CONSOLA

Cuando se analiza la entrada y ésta no tiene errores en consola se puede observar lo que se solicita

EJEMPLO DE DECLARACIÓN DE VARIABLES  
CONDICIONALES IF

```
Análisis completado
=== Archivo de prueba básico ===
===== Declaración de variables =====
Declaración explícita con tipo y valor
entero: 42
decimal: 3.14159
texto: Hola, mundo!
booleano: true
OK Declaración explícita con tipo y valor: correcto

Declaración explícita con tipo y sin valor
enteroSinValor: 0
decimalSinValor: 0
textoSinValor:
booleanoSinValor: false
OK Declaración explícita con tipo y sin valor: correcto

Errores de redeclaración
OK Errores de redeclaración: correcto

===== Asignación de variables =====
Asignación con tipo correcto
entero: 99
decimal: 9.9
texto: Texto modificado
booleano: false
OK Asignación con tipo correcto: correcto
```

Errores		Símbolos	
No		Descripción	

# 3. ERRORES

## Tipo Léxico

Ocurre cuándo no se encuentra el token en la gramática

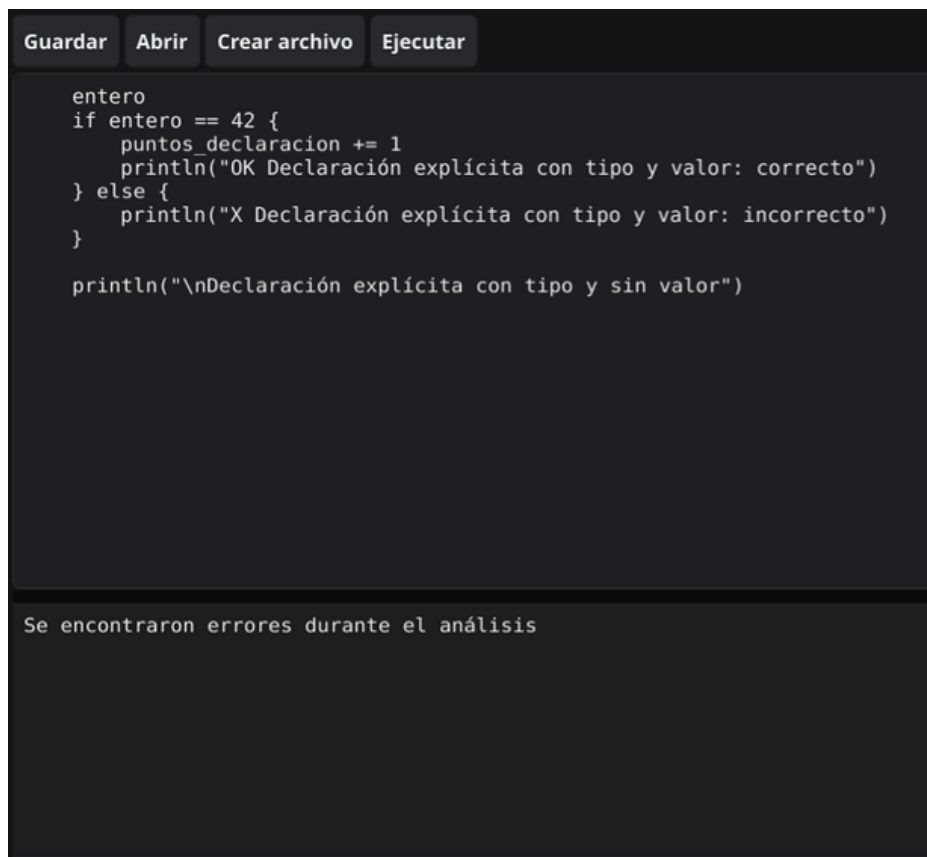
## Tipo Sintáctico

Ocurre cuando el orden de la entrada no es compatible o reconocida con la estructura de la gramática

## Tipo Semántico

Ocurre cuando la asignación, instrucción, o expresión de la entrada no es lógica

## Información de Editor y Consola



The screenshot shows a code editor with a dark background. At the top, there are four buttons: "Guardar", "Abrir", "Crear archivo", and "Ejecutar". Below the buttons, the code is as follows:

```
entero
if entero == 42 {
    puntos_declaracion += 1
    println("OK Declaración explícita con tipo y valor: correcto")
} else {
    println("X Declaración explícita con tipo y valor: incorrecto")
}

println("\nDeclaración explícita con tipo y sin valor")
```

Below the code editor, there is a console window with the following output:

```
Se encontraron errores durante el análisis
```

# Salida Tabla de errores

Errores		Símbolos			
No	Descripción	Línea	Columna	Tipo	
1	no viable alternative at input 'enteroif'	2	1	Error sintáctico	

## 4. TABLA DE SÍMBOLOS

### Información de la tabla

Hay 7 columnas en la tabla de información

1. **ID:** Indica el identificador o nombre
2. **Tipo de Símbolo:** Indica el conjunto al que pertenece el token es decir su lexema
3. **Tipo dato:** Indica el valor que adopta la variable
4. **Ámbito:** Muestra el ámbito en el que se declara el identificador
5. **Valor:** Información del “valor” de el id
6. **Línea:** Información de la línea de ubicación del token
7. **Columna:** Información de la columna de ubicación del token

### Gráfica de la tabla de símbolos al presionar “Símbolos”

Errores		Símbolos				
ID	Tipo símbolo	Tipo dato	Ambito	Valor	Línea	Columna
puntos	variable	int	global	0	1	0
puntosEntornos	variable	int	global	2	7	1
a	variable	int	global	10	10	1
b	variable	int	global	20	22	1

## Información de la tabla

## Gráfica del árbol CST

