UNIVERSIDAD DE SAN CARLOS DE GUATEMALA FACULTAD DE INGENIERÍA, ESCUELA DE CIENCIAS Y SISTEMAS

OLC1

Proyecto 1 - Manual Técnico

Diego Josue Guevara Abaj

Fecha: 22 de abril de 2024

Introducción

Bienvenido al manual de usuario de CompiScript+, un proyecto innovador surgido del curso de Organización de Lenguajes y Compiladores 1 de la Escuela de Ciencias y Sistemas de la Facultad de Ingeniería. Este manual está diseñado para brindarte una guía clara y concisa sobre el uso de esta herramienta, la cual tiene como objetivo principal aplicar los conocimientos adquiridos en análisis léxico y sintáctico para la creación de un intérprete sencillo, pero funcional.

Objetivo General: El objetivo principal de CompiScript+ es aplicar los conocimientos sobre la fase de análisis léxico y sintáctico de un compilador para la realización de un intérprete sencillo, con las funcionalidades principales para que sea funcional.

Objetivos Específicos:

Reforzar los conocimientos de análisis léxico, sintáctico y semántico para la creación de un lenguaje de programación.

Aplicar los conceptos de compiladores para implementar un proceso de interpretación de código de alto nivel.

Utilizar los conceptos de compiladores para analizar un lenguaje de programación y producir las salidas esperadas.

Aplicar la teoría de compiladores para la creación de soluciones de software.

Generar aplicaciones utilizando la arquitectura cliente-servidor.

Descripción General: El proyecto CompiScript+ surge como una iniciativa para crear un lenguaje de programación que permita a los estudiantes del curso de Introducción a la Programación y Computación 1 adquirir conocimientos sobre programación y las generalidades de

un lenguaje de programación. Este lenguaje será utilizado en las primeras prácticas de laboratorio del curso mencionado.

Como estudiante del curso de Compiladores 1, se te encomienda la tarea de llevar a cabo este proyecto, aprovechando tus altos conocimientos en análisis léxico, sintáctico y semántico. Este manual te acompañará en el proceso de uso de CompiScript+, proporcionándote las instrucciones necesarias para sacar el máximo provecho de esta herramienta.

Requisitos Mínimos:

Para poder utilizar CompiScript+ de manera adecuada, es necesario cumplir con los siguientes requisitos mínimos del sistema:

Sistema Operativo:

Ubuntu 22 o superior.

Software:

Node.js 20 o superior: Es necesario tener instalado Node.js en tu sistema para poder ejecutar CompiScript+. Puedes descargar la versión adecuada desde el sitio oficial de Node.js y seguir las instrucciones de instalación para Ubuntu.

Espacio en Disco:

Se recomienda tener al menos 100 MB de espacio libre en disco para la instalación y almacenamiento de archivos temporales generados por CompiScript+.

Memoria RAM:

Se recomienda disponer de al menos 2 GB de memoria RAM para un rendimiento óptimo al ejecutar CompiScript+ y realizar procesos de compilación e interpretación.

Navegador Web:

Se recomienda tener un navegador web actualizado, como Google Chrome, Mozilla Firefox o Microsoft Edge, para visualizar la documentación y posibles interfaces web relacionadas con CompiScript+.

Conexión a Internet (opcional):

Se recomienda disponer de una conexión a Internet para acceder a recursos adicionales, como actualizaciones, documentación en línea o posibles bibliotecas externas necesarias para el desarrollo con CompiScript+.

Es importante tener en cuenta que estos son requisitos mínimos y que el rendimiento del sistema puede variar dependiendo de la complejidad de los programas que se estén compilando o interpretando con CompiScript+. Se recomienda verificar periódicamente las actualizaciones de software y realizar las actualizaciones necesarias para garantizar un funcionamiento óptimo del sistema.

Interfaz de Usuario:

Funciones de Manejo de Eventos:

handleEditorDidMount: Se ejecuta cuando el editor de código está montado.

showValue: Envía el contenido del editor a un servidor para su interpretación y actualiza el estado con el resultado.

editor, getAst, getErrors, getSimbols: Cambian la vista entre el editor, el AST, los errores y los símbolos.

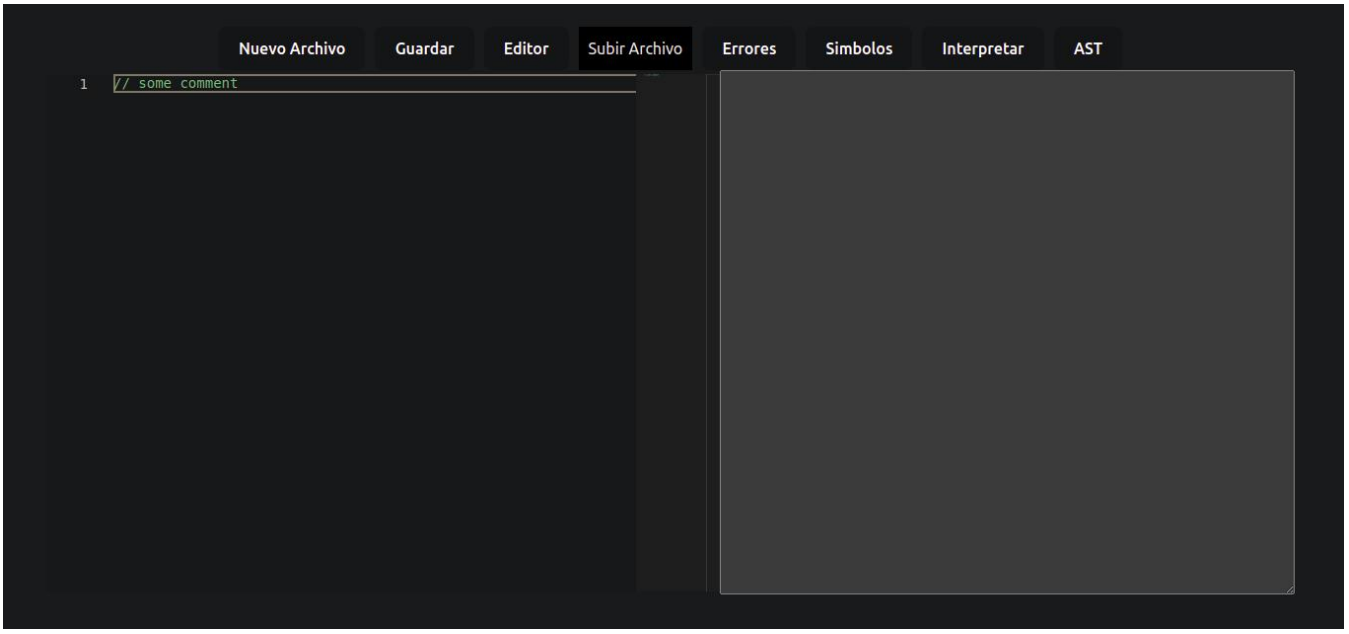
newArchive: Limpia el contenido del editor y reinicia los estados.

saveFile, openFile: Permiten guardar y abrir archivos respectivamente.

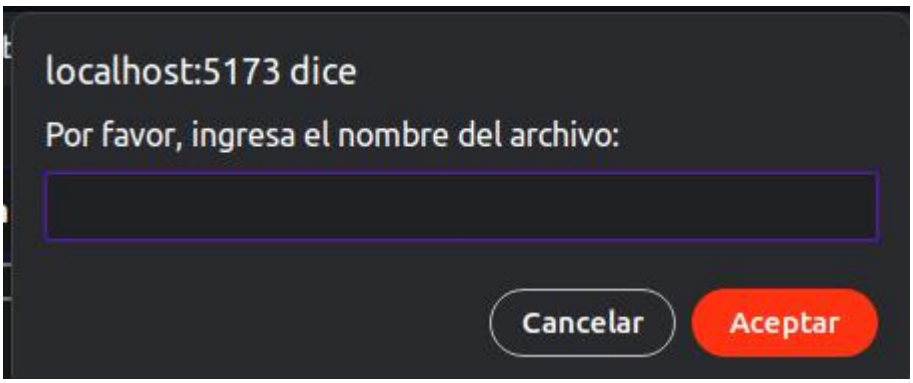
Interfaz de Usuario:

Botones para acciones como nuevo archivo, guardar, cambiar entre editor y otras vistas, subir archivo, interpretar, y visualizar AST.

Dependiendo de la vista seleccionada, se muestra el editor de código, el gráfico AST con funcionalidades de zoom, o los errores/símbolos en formato HTML.

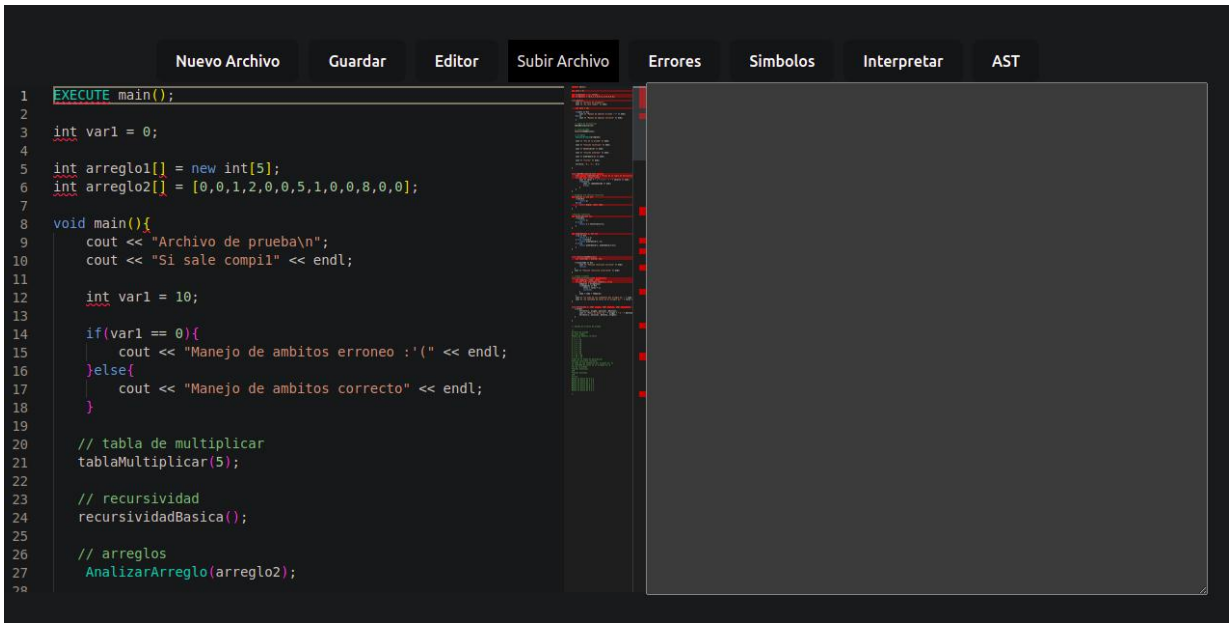


En el apartado para generar un nuevo archivo, se limpi la ventana y se pierde toda la información si esta no ha sido guardada antes.

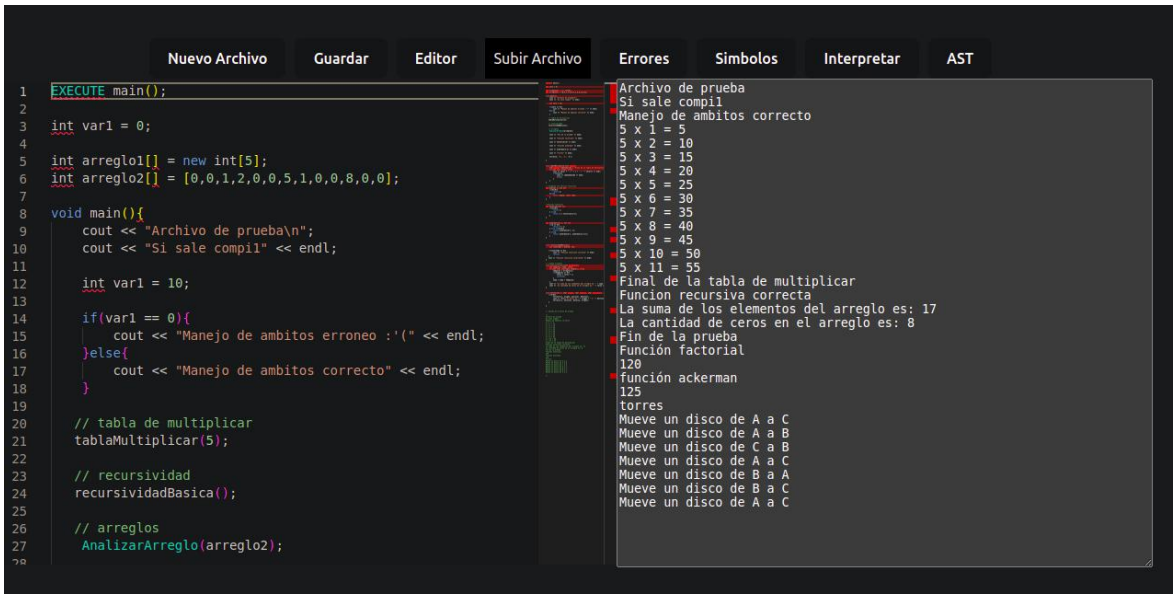


En el apartado para guardar archivo, se guardara lo contenido en el editor en ese momento. Ejemplos de archivos guardados. Cabe recalcar que estos ya traeran la extension “.sc” por defecto.

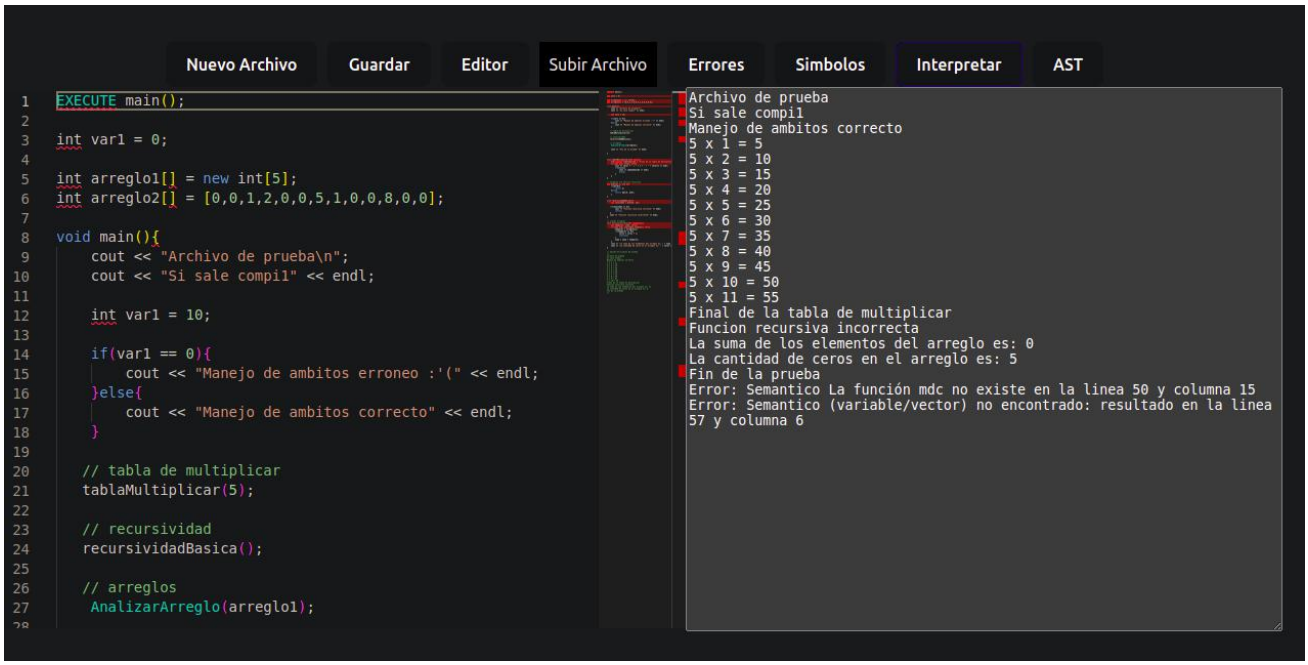
Al momneto de cargar un archivo este se colocara en el apartado del editor y sera manipulable por el usuario.



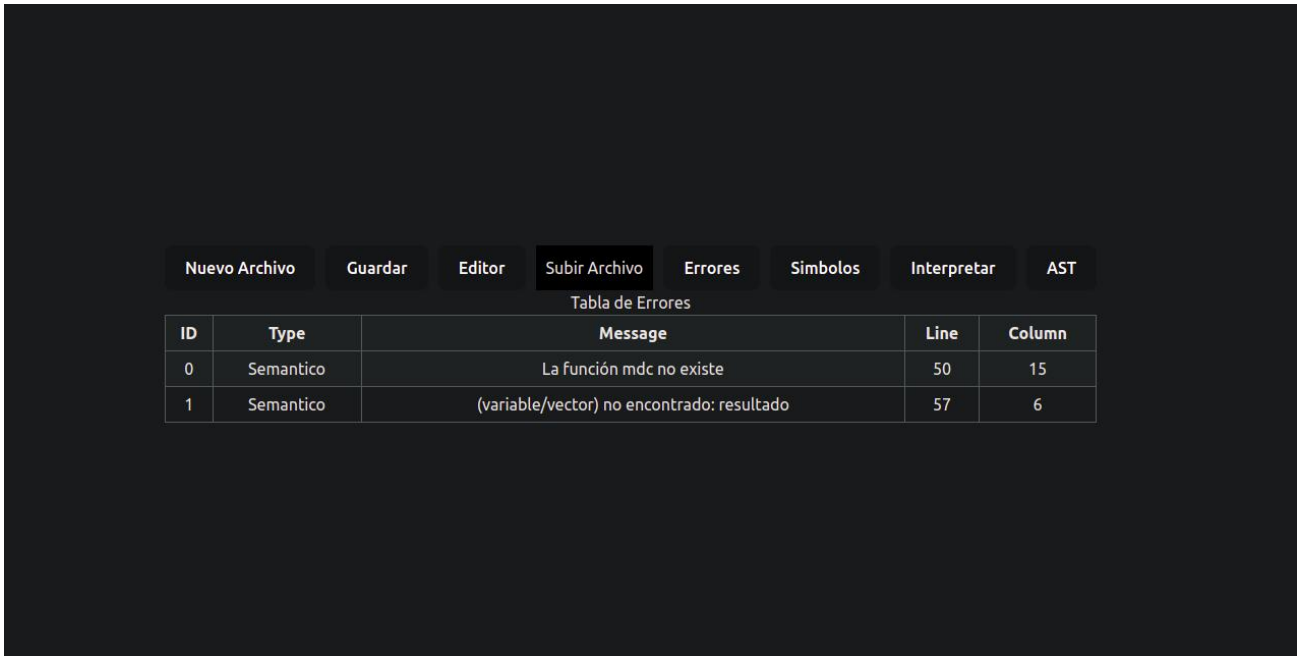
Al momento de darle a la instrucción de interpretar este sacara todos los resultados obtenidos por la ejecución del codigo



Si se llegaran a encontrar errores de tipo semantico o lexico se mostraran en la consola al finalizar la ejecución



Reportes de errores - Tabla de Simbolos



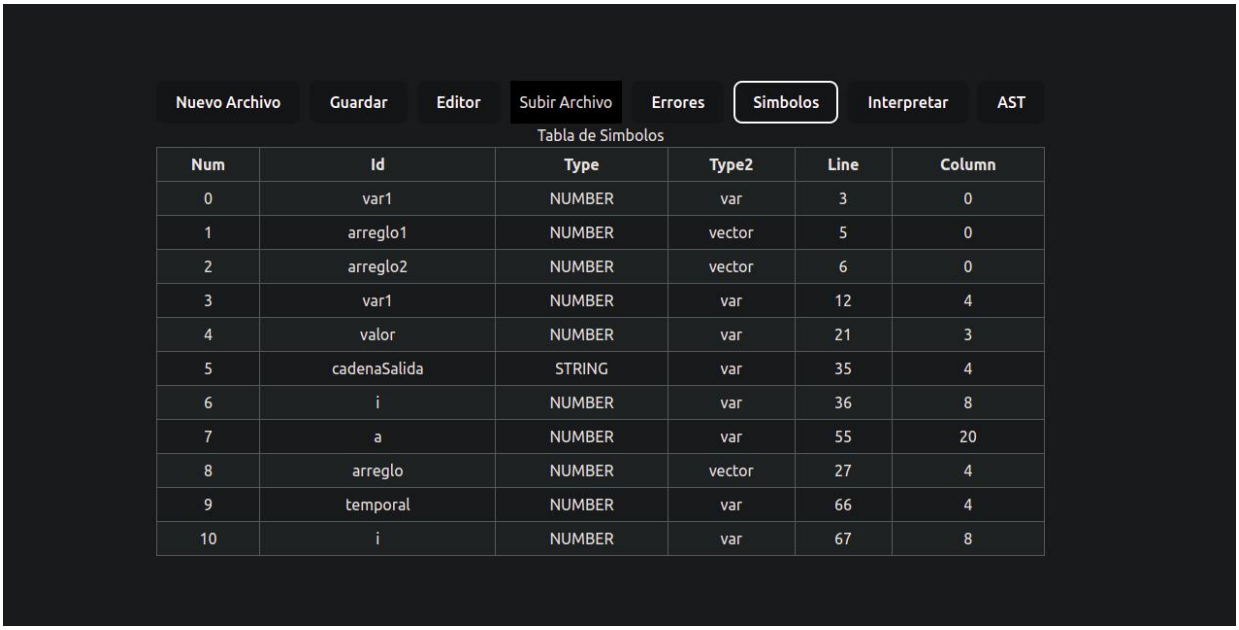


Tabla de Simbolos

Num	Id	Type	Type2	Line	Column
0	var1	NUMBER	var	3	0
1	arreglo1	NUMBER	vector	5	0
2	arreglo2	NUMBER	vector	6	0
3	var1	NUMBER	var	12	4
4	valor	NUMBER	var	21	3
5	cadenaSalida	STRING	var	35	4
6	i	NUMBER	var	36	8
7	a	NUMBER	var	55	20
8	arreglo	NUMBER	vector	27	4
9	temporal	NUMBER	var	66	4
10	i	NUMBER	var	67	8

El apartado de AST mostrara el diagrama generado por el codigo correspondiente utilizado.