



Universidad de San Carlos de Guatemala
Facultad de Ingeniería
Escuela de Ciencias y Sistemas
Organización de Lenguajes y Compiladores 1
Catedrático: Ing. Kevin Adiel Lajpop Ajpacaja
Tutor académico: Moises Gonzalez Fuentes

SECCION C

Segundo semestre 2022

MANUAL DE USUARIO

Proyecto 1

Pseudo-Parser

Nombre: Moises David Maldonado de León

Carné: 202010833

17/09/2022

Objetivos del sistema

El siguiente sistema va dirigido para que todo nuevo personal que no conoce los lenguajes de Python y Golang. Se diseñó como parte de una solución informática que permita ser una aplicación de traducción con la facultad de traducir el pseudocódigo ingresado a uno de estos lenguajes de programación de una manera más sencilla y amigable a la vista del usuario.

Información del sistema

Este sistema posee un analizador léxico y sintáctico, implementados mediante las herramientas JFLEX y CUP respectivamente para el manejo de la traducción del pseudocódigo, a su vez, genera una tabla de errores (tanto léxicos como sintácticos) detectados con los datos de fila y columna para una mayor facilidad de detección y corrección. La aplicación permite mostrar el diagrama de flujo y como parte adicional, se puede visualizar el árbol de análisis sintáctico, así como las traducciones y demás reportes. Finalmente, el sistema permite acceder a la documentación del sistema para una mejor comprensión del funcionamiento y la correcta ejecución del sistema.

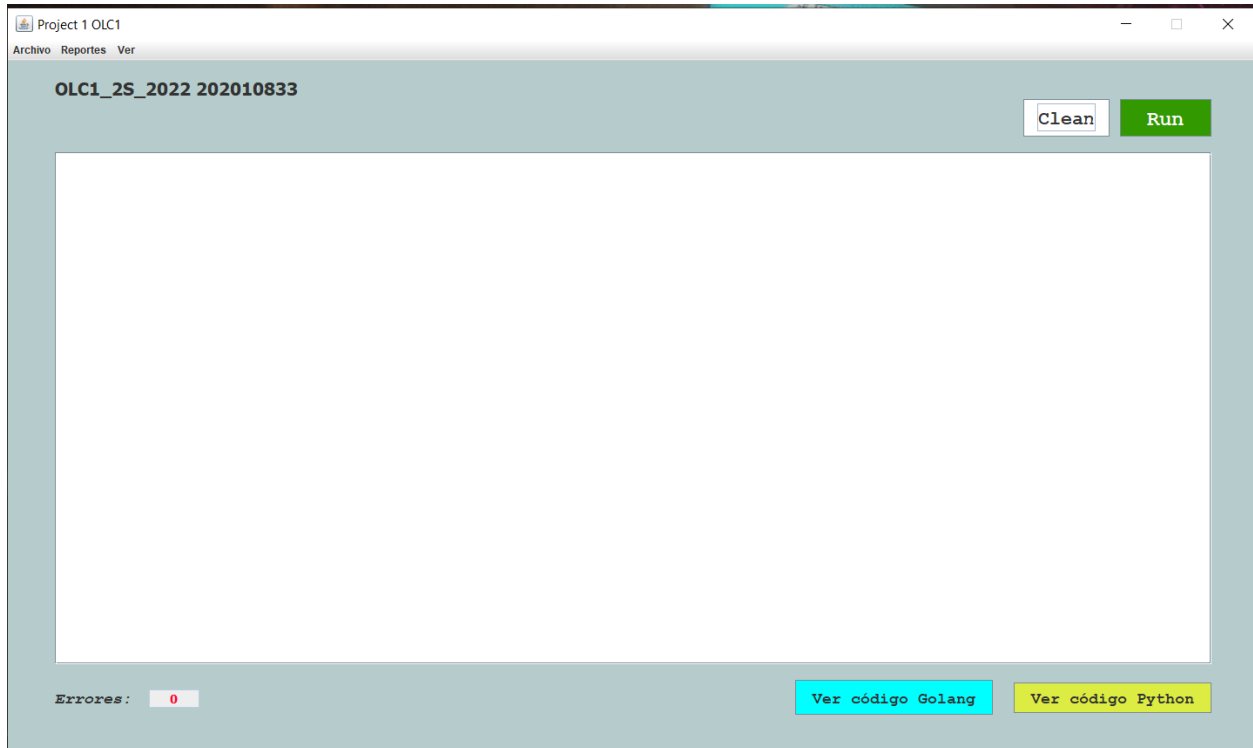
Requisitos del Sistema

Procesador	Intel Core 2 Duo 2 GHz o superior
Memoria RAM	De 2GB en adelante
Espacio en disco duro	50 mb mínimo
Sistema operativo	Windows o Mac OS
Conexión a internet	No es necesaria
Herramientas	<p>Necesita el archivo ejecutable (/dist/ Proyecto_OLC1 .jar) y demás componentes</p> <p>Se requiere tener las siguientes librerías:</p> <ul style="list-style-type: none">• java-cup-11b.jar• java-cup-11b-runtime.jar• jflex-full-1.7.0.jar• Graphviz

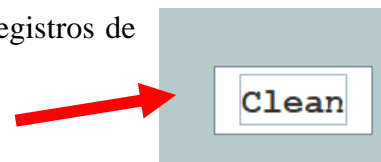
Interfaz gráfica

Menú principal

Se despliega un menú principal donde se identifican 4 botones y una barra de menús.



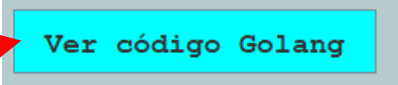
1. **Clean:** Este botón limpia el área de texto, así como los registros de errores y el árbol sintáctico.



2. **Run:** Este botón permite al usuario ejecutar el flujo del programa.



3. **Ver código Golang:** Este botón despliega el explorador de archivos para elegir una ruta de guardado y el archivo se creará con extensión Golang (.go).



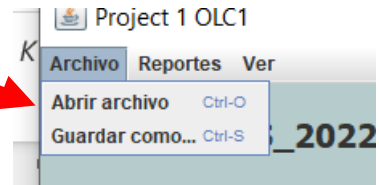
Ver código Golang

4. **Ver código Python:** Este botón despliega el explorador de archivos para elegir una ruta de guardado y el archivo se creará con extensión Python (.py).



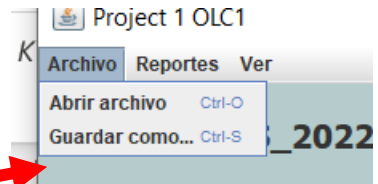
Ver código Python

5. **Abrir archivo:** Permite seleccionar un archivo de extensión OLC (.olc) de su directorio de archivos y trasladar el contenido al área de texto.

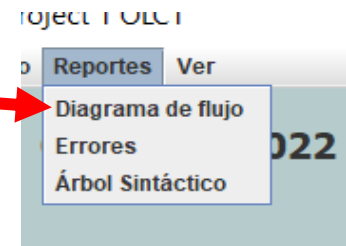


6.

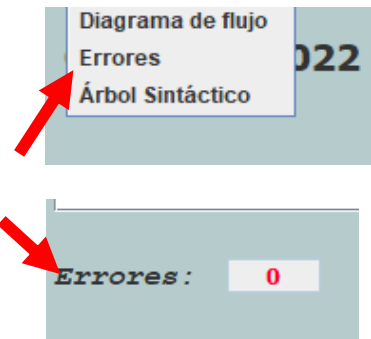
7. **Guardar como...:** Esta opción permite desplegar el explorador de archivos para seleccionar la ruta de guardado, así como el nombre que se le dará. Se creará un archivo de extensión OLC (.olc).



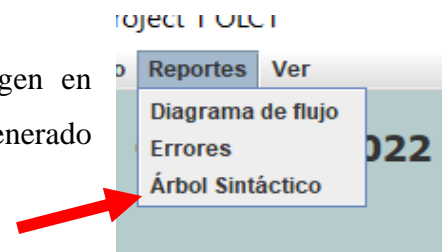
8. **Diagrama de flujo:** Esta opción permite visualizar el diagrama de flujo del pseudocódigo analizado.



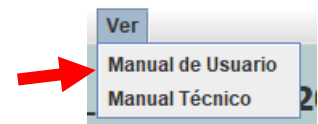
9. **Errores:** Esta opción despliega una ventana emergente con una tabla dónde se visualizan los errores encontrados y su respectiva descripción, además, se podrá visualizar la cantidad de errores en la ventana principal (etiqueta de errores).



10. **Árbol sintáctico:** Esta opción permite genera una imagen en formato png dónde se muestra todo el árbol sintáctico generado luego de su posterior análisis.



11. **Manual de Usuario:** Esta opción despliega el manual de usuario de la aplicación.



12. **Árbol sintáctico:** Esta opción despliega el manual técnico de la aplicación.



Flujo de funcionalidades del sistema

Paso 1. Escritura del pseudocódigo

Ingresa el pseudocódigo en el área de texto o de clic en el menú archivo, luego al submenú abrir archivo y proceda a elegir el archivo (extensión .olc).

-Descripción del lenguaje válido

Tipos de datos reconocidos:

- Numero: Números positivos enteros o decimales.
- Cadena: Conjunto de caracteres dentro de comillas dobles.
- Boolean: Tipo de dato verdadero o falso.
- Caracter: Un carácter dentro de una comilla simple. Es posible ingresar el código ascii (únicamente los ascii de caracteres de alfabeto).

Operaciones básicas:

- Suma: $\langle \text{operando1} \rangle + \langle \text{operando2} \rangle$.
- Resta: $\langle \text{operando1} \rangle - \langle \text{operando2} \rangle$
- Multiplicación: $\langle \text{operando1} \rangle * \langle \text{operando2} \rangle$
- División: $\langle \text{operando1} \rangle / \langle \text{operando2} \rangle$
- Potencia: $\langle \text{operando1} \rangle \text{ potencia } [\langle \text{operando2} \rangle]$
- Módulo: $\langle \text{operando1} \rangle \text{ modulo } \langle \text{operando2} \rangle$.
- Paréntesis: $(\langle \text{conjunto de operandos} \rangle)$

Operadores Relacionales:

- Mayor: $\langle \text{Expresion} \rangle \text{ Mayor } \langle \text{Expresion} \rangle$.
- Menor: $\langle \text{Expresion} \rangle \text{ Menor } \langle \text{Expresion} \rangle$
- Mayor_o_igual: $\langle \text{Expresion} \rangle \text{ Mayor_o_igual } \langle \text{Expresion} \rangle$
- Menor_o_igual: $\langle \text{Expresion} \rangle \text{ Menor_o_igual } \langle \text{Expresion} \rangle$
- Es_Igual: $\langle \text{Expresion} \rangle \text{ Es_Igual } [\langle \text{Expresion} \rangle]$
- Es_Diferente: $\langle \text{Expresion} \rangle \text{ Es_diferente } \langle \text{Expresion} \rangle$.

Operadores Lógicos:

- or: <Expresion> or < Expresion >.
- and: < Expresion > and < Expresion >
- not: not < Expresion >

Global:

- Inicio: Palabra reservada de inicio.
- Cuerpo: Espacio donde se escribe el pseudocódigo.
- Fin: Palabra reservada de fin.

Comentarios:

- //Comentarios: Comentario de una línea.
- /*comentario*/: Comentario de multilínea.

Declaración:

- Ingresar: Palabra reservada.
- _ID_: Lista de uno o más nombres de variable encerrado entre guiones bajos y separados por comas.
- como: Palabra reservada.
- TIPO: Debe colocar el tipo de dato.
- Con_valor: Palabra reservada.
- Expresión: Puede ser expresión aritmética u otro tipo de dato permitido.
- Cerrar: Debe cerrar con punto y coma la sentencia (;)

Asignacion:

- _ID_: Lista de uno o más nombres de variable encerrado entre guiones bajos y separados por comas.
- Asignación: Símbolo (->)
- Expresión: Puede ser expresión aritmética u otro tipo de dato permitido.

Condicional Si:

O_si puede repetirse cero, una o más veces, el apartado de_lo_contrario es opción opcional. Instrucciones puede generar cualquier sentencia o ciclo descritos en este documento.

```
si <condición>
    <instrucciones>
o_si <condición_nueva> entonces
    <instrucciones>
o_si <condición_nueva> entonces
    <instrucciones>
de_lo_contrario
    <instrucciones>
fin_si
```

Selección múltiple:

Este tipo de condición permite ejecutar una lista de instrucciones en base a un valor ingresado, existen varias opciones posibles y cuando el valor coincida con una de las opciones se ejecuta un conjunto de instrucciones. Existe una palabra reservada “de_lo_contrario” para ejecutar automáticamente cuando la lista de opciones no se cumple, pero es de forma opcional.

segun <valor> hacer

```
¿ <valor 1> ? entonces
    <instrucciones para valor 1>
¿ <valor 2> ? entonces
    <instrucciones para valor 2>
¿ <valor 3> ? entonces
    <instrucciones para valor 3>
de_lo_contrario entonces
```

fin_segun

<instrucciones por default>

Ciclo Para:

Este ciclo ejecuta un conjunto de instrucciones, con un límite de repeticiones. Es necesario ingresar un valor inicial y también un valor final, el valor final es el que le indica al ciclo, en momento para terminar de realizar las repeticiones. Otro de los elementos necesarios en este ciclo es el número de pasos que realizará entre cada repetición. Cuando el ciclo no tiene definido el número de pasos, se tomará como defecto el incremento en 1. Es posible que la lista de instrucciones esté vacía.

```
//Estructura de ciclo para con salto no definido
para <variable> -> <valor inicial> hasta <valor final> hacer
    <instrucciones>
fin_para

para <variable> -> <valor inicial> hasta <valor final> hacer
    //null
fin_para

//Estructura de ciclo para con salto definido
para <variable> -> <valor inicial> hasta <valor final> con incremental
    <Valor del salto> hacer
    <instrucciones>
fin_para
```

Ciclo mientras:

Este ciclo ejecuta un conjunto de instrucciones sin límite definido. Para poder realizar una repetición es necesario que exista una condición. Es posible que la lista de instrucciones esté vacía.

```
mientras <condicion> hacer
    <intrucciones>
fin_mientras
```

```
mientras <condicion> hacer
    //null
fin_mientras
```


Ciclo Repetir hasta:

Este ciclo ejecuta un conjunto de instrucciones sin límite definido. A diferencia del ciclo anterior, este ciclo realiza una única repetición sin restricción. Para poder realizar las demás repeticiones es necesario que exista una condición. Es posible que la lista de instrucciones esté vacía.

```
repetir
    <instrucciones>
hasta_que <condición>
```

```
repetir
    //null
hasta_que <condición>
```

Retorno:

- retornar: Palabra reservada.
- Valor: Puede ser expresión aritmética, condición o número
- Cerrar: Debe cerrar la sentencia con punto y coma (;).

Método:

Esta instrucción permite agrupar un conjunto de instrucciones y asignarles un nombre para identificarlo dentro del contenido del archivo. No necesita una instrucción de “Retorno” y si en caso es reconocido este tipo de instrucción, debe reportar dicho error. Es posible agregar parámetros al método, estos parámetros tendrán definido el tipo de dato y su respectivo nombre. Cada uno de los parámetros estará separado por un carácter coma:

```
metodo <nombre> con_parametros (<lista de parametros>)
    <instrucciones>
fin_metodo
```

```
metodo <nombre>
    <instrucciones>
fin_metodo
```

Funciones:

Esta instrucción permite agrupar un conjunto de instrucciones y asignarles un nombre para identificarlo dentro del contenido del archivo. Esta instrucción si es posible reconocer una instrucción de “Retorno” en su estructura. Es posible agregar parámetros al método, estos parámetros tendrán definido el tipo de dato y su respectivo nombre. Cada uno de los parámetros estará separado por un carácter coma:

```
funcion <nombre> <tipo dato> con_parametros (<lista de parámetros>)
    <instrucciones>
fin_funcion
```

```
funcion <nombre> <tipo dato>
    <instrucciones>
fin_funcion
```

Llamada de funciones y métodos:

Este tipo de instrucción realiza la ejecución de un método o función, para poder realizarlo es necesario ingresar el identificador de la función o método y la lista de parámetros necesarios. En <Lista de parámetros> los parámetros están separados por el carácter coma.

```
//ejecutar sin parámetros  
ejecutar <identificador>;  
  
//ejecutar con parámetros  
ejecutar <identificador>(<Lista de parámetros>);
```

Impresión:

Esta instrucción muestra el contenido de una expresión o valor de una variable. Para poder utilizarla es necesario una expresión o valor de una variable. Al terminar de realizar la impresión se genera un salto de línea por defecto.

```
imprimir <expresión>; //impresión sin salto de línea  
imprimir_nl <expresión>; //impresión con salto de línea
```

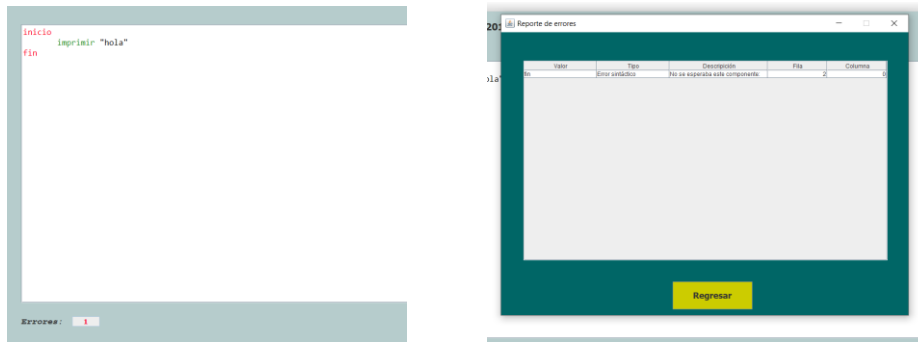
Observaciones:

- ✓ El lenguaje no hace distinción entre mayúsculas y minúsculas.
- ✓ Asegúrese de escribir correctamente la sintaxis.
- ✓ *Las palabras reservadas se mostrarán de otro color si se escriben en minúsculas:*

```
inicio  
    imprimir "hola";  
fin
```

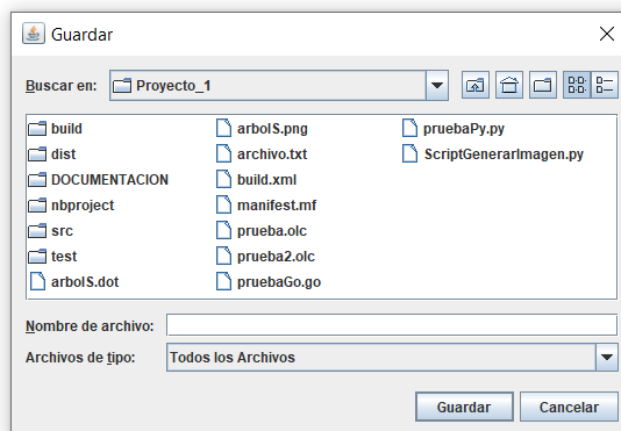
Paso 2.

De clic en “Run” para iniciar con el análisis, de existir errores se mostrará en la etiqueta errores y puede consultar las especificaciones en el menú Reportes y su respectivo submenú Errores.



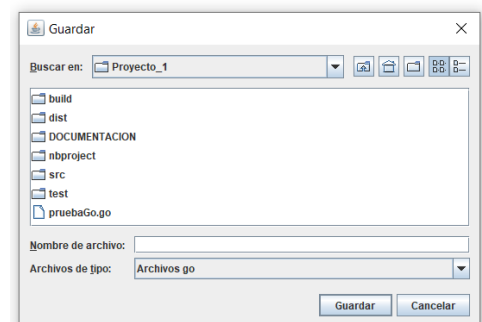
Paso 3 (Opcional).

De clic en el menú archivo y luego de clic en guardar como para escoger una ruta y un nombre para poder conservar su archivo de pseudocódigo.



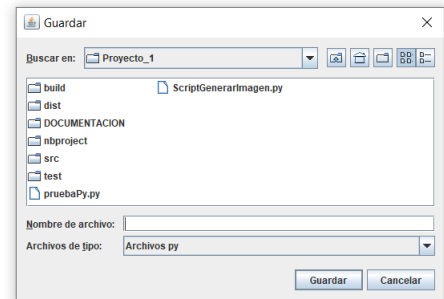
Paso 4 (Opcional).

De clic en el botón “Ver código Golang” para guardar su archivo .go.



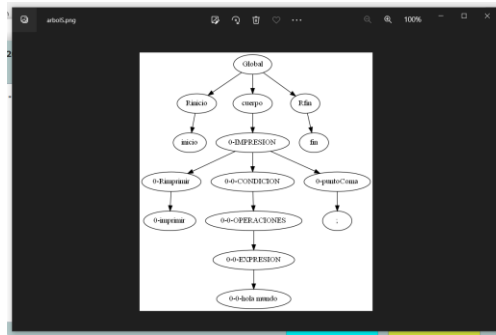
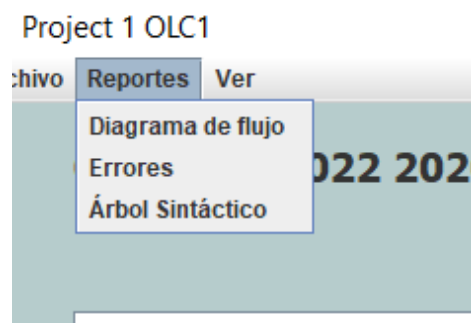
Paso 5 (Opcional).

De clic en el botón “Ver código Python” para guardar su archivo .py.



Paso 6 (Opcional).

De clic en el menú reportes y seleccione la opción de Árbol Sintáctico. Se generará la imagen y se mostrará automáticamente. La ruta de la imagen será en el directorio del código fuente.



nbproject	17/8/2022 14:15	Carpeta de archivos	
src	31/8/2022 09:24	Carpeta de archivos	
test	17/8/2022 14:23	Carpeta de archivos	
arbolS.dot	17/9/2022 23:49	Plantilla de Micros...	1 KB
arbolS.png	17/9/2022 23:49	Archivo PNG	61 KB
archivo.txt	17/9/2022 23:47	Documento de tex...	1 KB
build.xml	17/8/2022 14:15	Documento XML	4 KB
manifest.mf	17/8/2022 14:15	Archivo MF	1 KB

Paso 7 (Opcional recomendado).

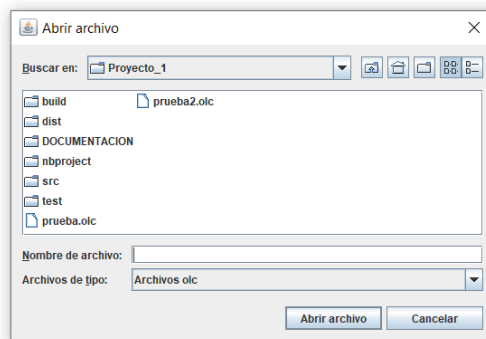
Se recomienda al usuario que después de ingresar un pseudocódigo de clic en el botón “Clean” para llevar un orden en el árbol sintáctico y las respectivas traducciones.

Paso 8 (Opcional).

Se recomienda al usuario que después de ingresar un comando de clic en el botón “Limpiar Log de Errores” para llevar un mejor control de los errores ingresados.

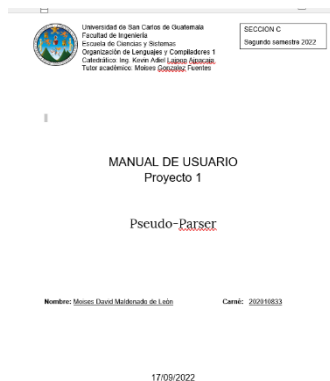
Paso 9 (Opcional).

Vaya al menú archivo, de clic en abrir como y seleccione un archivo .olc para poder ver su contenido.




Paso 10 (opcional).

Vaya al menú ver y de clic en “Manual de usuario”. Esto abrirá el manual de usuario de manera automática para que el usuario pueda conocer las funcionalidades de los botones, así como el uso correcto del sistema.



Paso 11 (opcional).

Vaya al menú ver y de clic en “Manual técnico”. Esto abrirá el manual de usuario de manera automática para que el usuario pueda conocer la lógica detrás del sistema y la construcción previa del programa (Creación de expresiones regulares y AFD para cada token.)

	Universidad de San Carlos de Guatemala Facultad de Ingeniería Escuela de Ciencias y Sistemas Organización de Lenguajes y Compiladores 1 Catedrático: Ing. Kevin Adiel Laipon Alpacaia Tutor académico: Moises Gonzalez Fuentes	SECCION C Segundo semestre 2022
---	---	------------------------------------

MANUAL TÉCNICO

Proyecto 1

Pseudo-Parser

Nombre: Moises David Maldonado de León **Carné:** 202010833

Guatemala, 17 de septiembre de 2022