Diseño de Compiladores (IS-913)

II PAC 2023

Grupo No. 6

Delta Nova

Calculadora de Operaciones

Elaborado por:

Kenet Francisco Orellana Meza	**20141011708**
Marvin Dionicio Martínez Aguilera	**20161004203**
Luis Enrique Verde Sánchez	**20171031756**

Catedrático:

Josian Francisco Cáceres Suazo

Fecha:

14 de agosto de 2023

ÍNDICE

INTRODUCCIÓN

A continuación, se detallan las diferentes características y los procesos involucrados en el desarrollo de un traductor de expresiones aritméticas básico que por medio de una entrada digitada por un usuario nos permita conocer y analizar los pasos que sigue un traductor.

Es importante mencionar que el analizador léxico y el sintáctico están vinculados ya que dentro del proceso de compilación el analizador léxico interviene primero donde este toma la entrada digitada por el usuario y divide cada carácter en tokens o unidades léxicas significativas para el compilador, estos tokens son las unidades mínimas que puede reconocer el traductor.

Por otro lado, el analizador sintáctico se encarga de comparar las instrucciones ingresadas con las reglas establecidas en la gramática definida previamente para producir una salida que cumpla con estas reglas del compilador.

Tecnologías usadas para la creación de una Calculadora de Operaciones:

- **PLY:** Generador de analizadores léxicos y sintácticos.
- **Python 3:** Lenguaje de programación interpretado de alto nivel.
- **Visual Studio Code (VS Code):** Editor de código ligero, pero poderoso.

OBJETIVOS

GENERALES & ESPECÍFICOS

Generales:

- Desarrollar un traductor básico de operaciones aritméticas definido por el usuario por medio de la implementación de una interfaz gráfica de usuario (GUI) de manera que dicha operación realice un análisis léxico y sintáctico de la cadena entrante y muestre el resultado.

Específicos:

- Por medio de la biblioteca PLY de Python implementar un analizador léxico que permita identificar los tokens de la expresión aritmética ingresada.
- Por medio de la biblioteca PLY de Python implementar un analizador sintáctico que utilice la gramática definida previamente para obtener un resultado.
- Proporcionar una interfaz gráfica de usuario que le permita a los usuarios ingresar expresiones aritméticas básicas y visualicen una respuesta correcta.

TOKENS, PATRONES y LEXEMAS

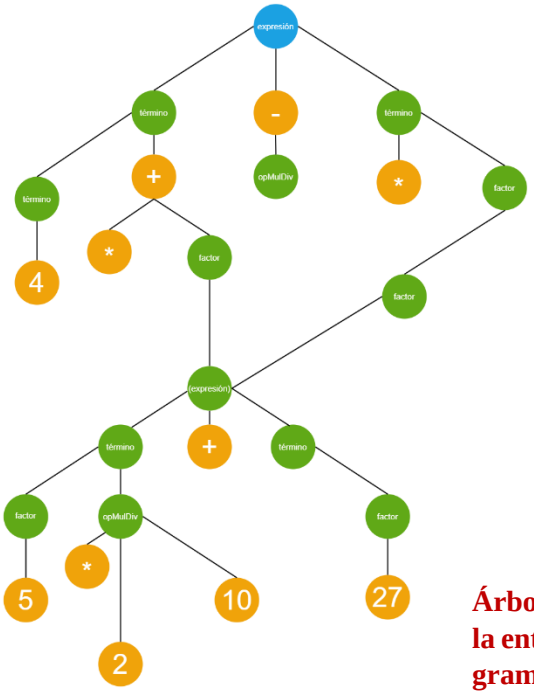
Token	Lexema	Patrón
Entero – int	1, 2, 3, 4, 5, 6, 7… 20, 21… n	Conjunto de números naturales.
Decimal – dec	10.2, 34.8, 45.5… 50.252… n	Compuesto de unidades enteras (números naturales) y de una fracción decimal.
Operadores – op	○ Suma: '+' ○ Resta: '-' ○ Multiplicación: '*' ○ División: '/'	Aplicado en una operación que incluya dos enteros, dos decimales o ambos.
Paréntesis – parens	○ De apertura: '(' ○ De cierre: ')'	Utilizado de acuerdo a su jerarquía.
Operación aritmética	5+9*(4-2)/2	Función sobre una tupla y que obtiene un resultado, aplicando reglas preestablecidas sobre la tupla.

REGLAS DE PRODUCCIÓN

Una regla de producción es un conjunto de reglas o procedimientos para llevar a cabo una tarea que especifica una sustitución de uno o varios símbolos que se puede realizar de forma recursiva para generar nuevas secuencias de símbolos.

Para efectos de nuestra **Calculadora de Operaciones** la gramática que hemos definido contempla las operaciones aritméticas básicas como ser suma, resta, multiplicación y división, junto con el uso de paréntesis para agrupar las operaciones por orden de jerarquía.

expresión → expresión opSumRes término | término
opSumRes → + | -
término → término opMulDiv factor | factor
*opMulDiv → * | /*
factor → (expresión) | número
número → entero '.' entero | entero
número → 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9



Árbol sintáctico generado a partir de la entrada 4+2*(5+10)-27 en base a la gramática previamente definida.

REGLAS SEMÁNTICAS

p_calculo(p)

Se utiliza para definir la estructura de la expresión que se está analizando y cómo se debe generar la salida a partir de ella.

En este caso, la regla cálculo puede tener dos opciones: una expresión o vacío. Si la regla cálculo se reduce a una expresión, entonces la función correr() se utiliza para evaluar la expresión y el resultado se agrega a la cadena de salida. Si la regla cálculo se reduce a vacío, significa que el usuario no ha ingresado nada y la cadena de salida simplemente se establece en una cadena vacía.

La variable p es una tupla que contiene los elementos asociados a la regla sintáctica cálculo. En este caso, p[1] se refiere a la expresión analizada por el analizador sintáctico. La cadena de salida se construye concatenando la expresión analizada, las operaciones involucradas y el resultado obtenido al evaluar la expresión.

---

p_expresion_ENTERO_DECIMAL(p)

Se encarga de definir el comportamiento para cuando el analizador léxico encuentra un token que corresponde a un número entero o decimal.

Esto significa que la regla semántica p_expresion_ENTERO_DECIMAL acepta una expresión que consiste en un número entero o decimal, que son representados por los tokens ENTERO o DECIMAL respectivamente.

La acción semántica que realiza esta regla es asignar el valor del token correspondiente al atributo p[0]. En este caso, como la regla sólo contiene un token, el valor del atributo p[0] será igual al valor del token (p[1]), lo que significa que el número entero o decimal será retornado como resultado de la evaluación de la expresión.

---

p_expresion(p)

Esta función define la regla semántica para una expresión matemática que involucra una operación binaria, como multiplicación, división, suma o resta. Cada una de las producciones de la regla define cómo se deben combinar dos expresiones para formar una nueva expresión.

En todas las producciones, se almacena el operador, la expresión izquierda y la expresión derecha en una tupla, que se asigna a p[0]. Esto permite que la expresión completa se representa como un árbol sintáctico, con los operadores en los nodos internos y las expresiones en los nodos hoja.

---

p_expresion_par(p)

Define una regla gramatical para manejar una expresión que está contenida entre paréntesis. La regla específica que debe haber un paréntesis izquierdo '(' seguido de una expresión, seguido de un paréntesis derecho ')'. La expresión dentro de los paréntesis puede ser cualquier otra expresión definida en las reglas gramaticales.

Cuando se encuentra una expresión que se ajusta a esta regla gramatical, se guarda el valor de la expresión sin los paréntesis (es decir, p[2] en la regla gramatical) en p[0].

Esto se hace para que la expresión dentro de los paréntesis pueda ser tratada como una expresión completa en otras reglas gramaticales que la contengan.

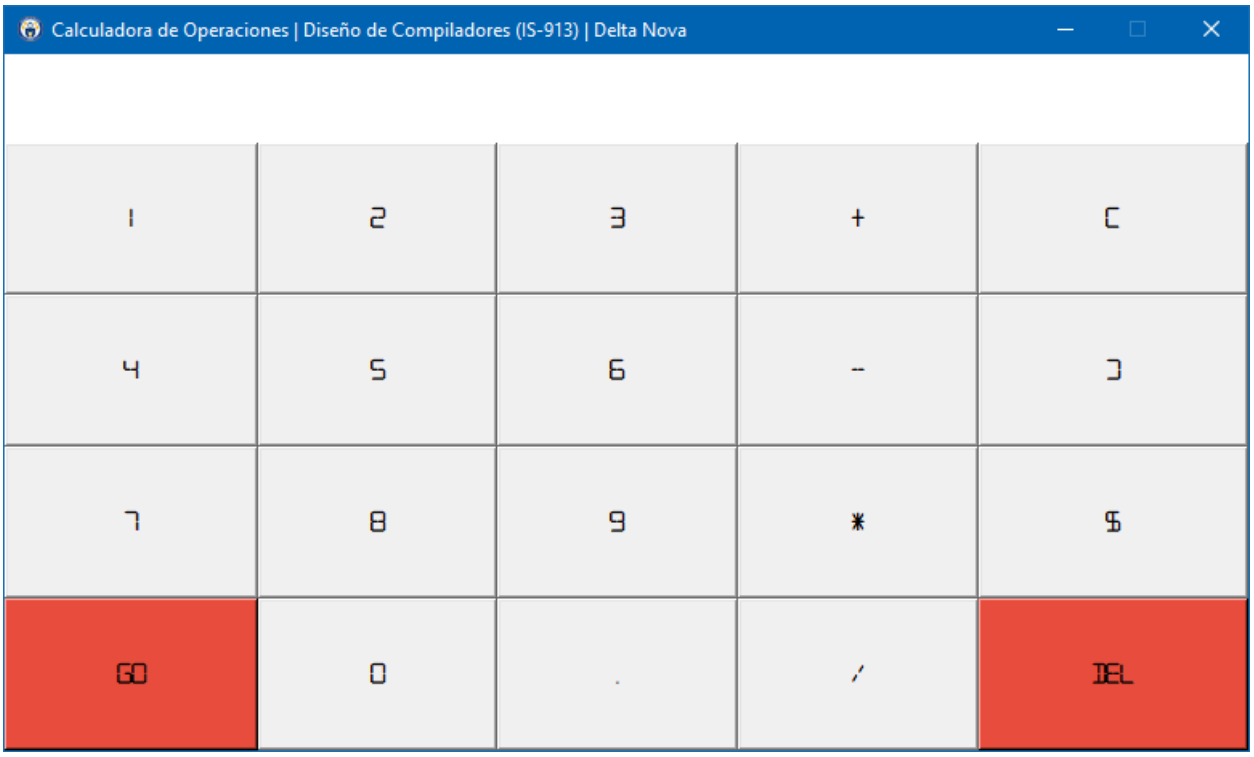
---

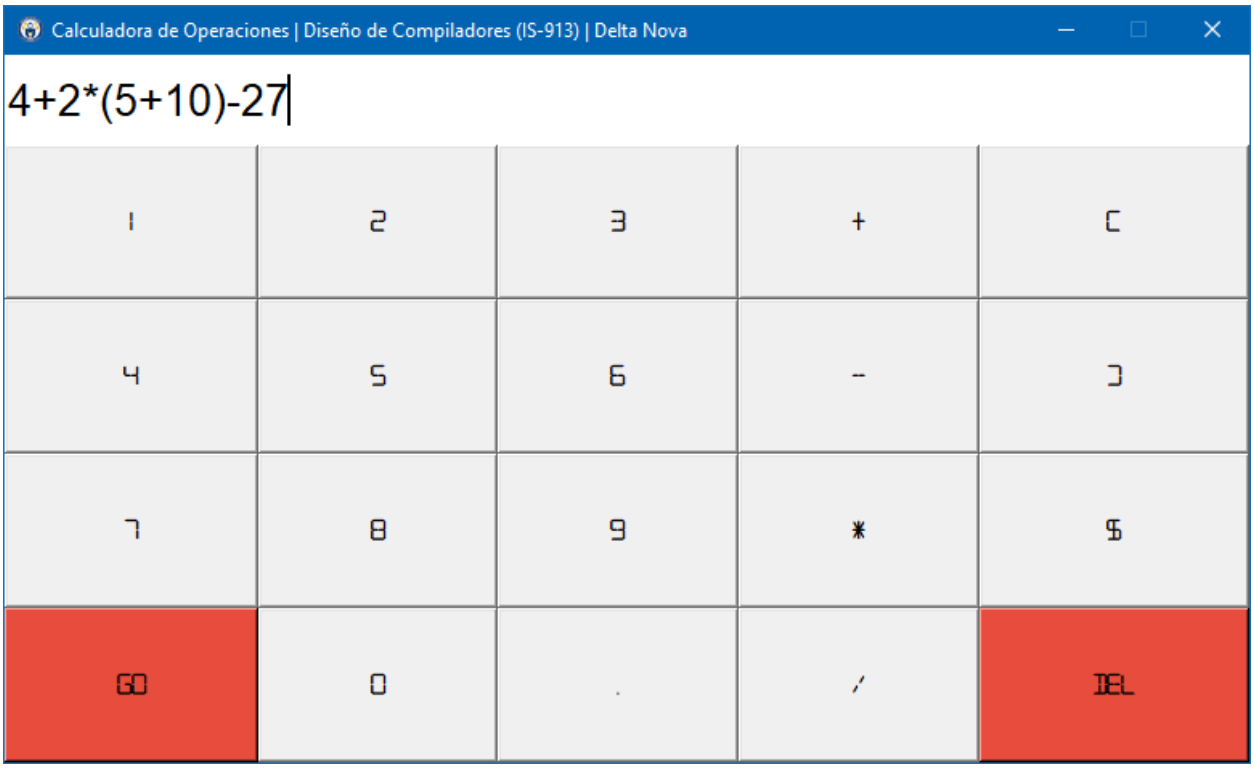
p_vacio(p)

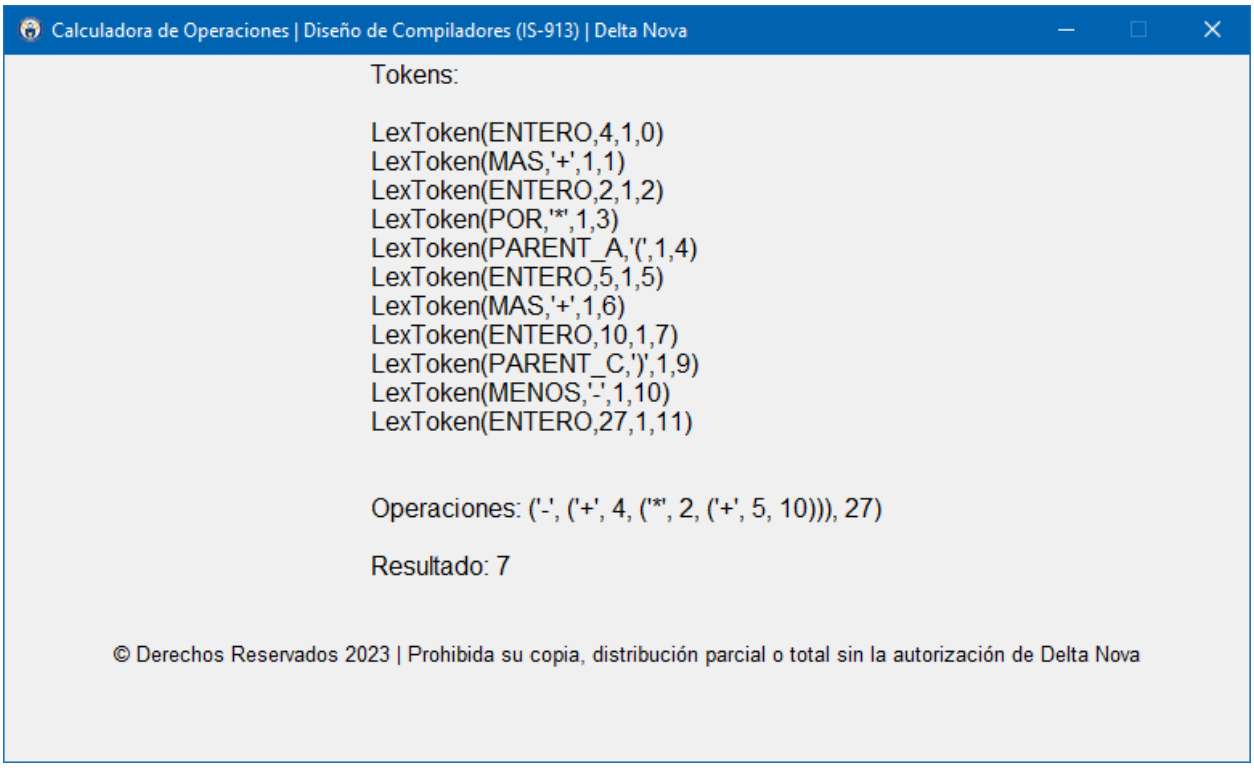
Esta es una regla gramatical para la producción vacío, que representa una expresión vacía. En otras palabras, esta regla se utiliza cuando no se ingresa ninguna expresión matemática por parte del usuario y se evalúa como None.

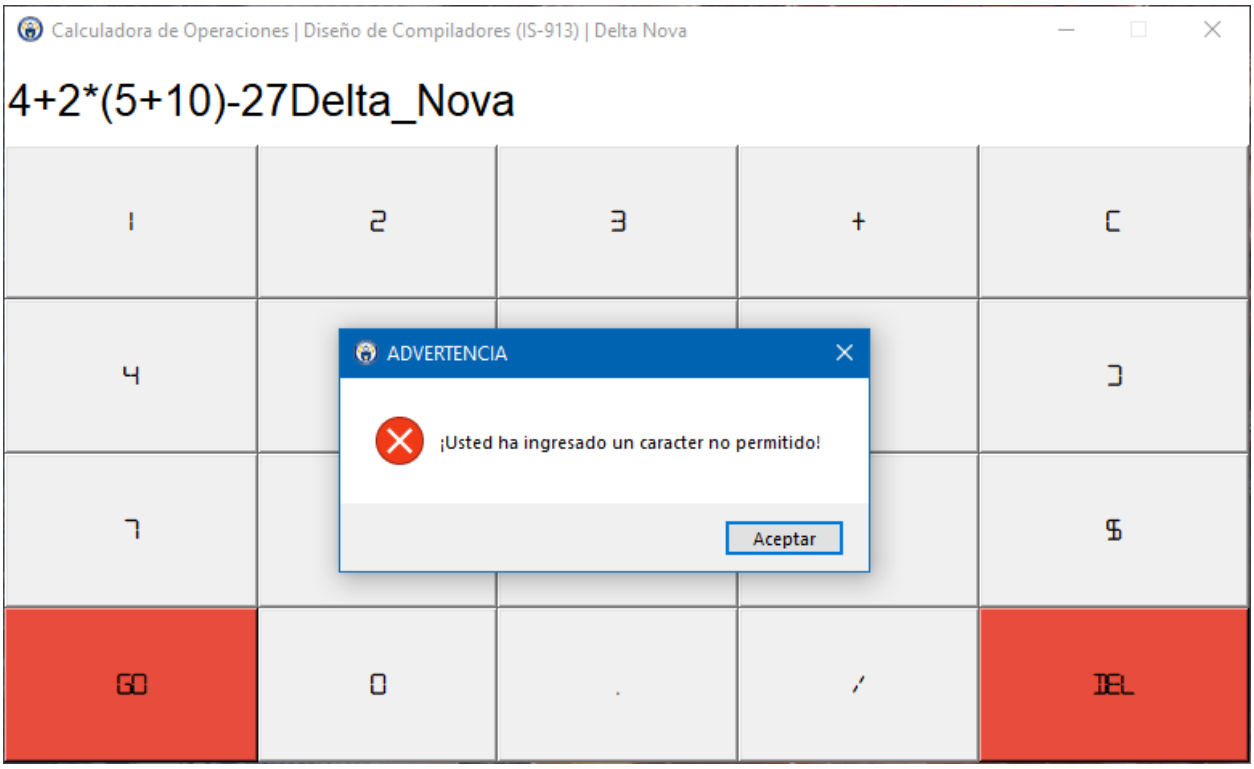
En términos de sintaxis, esta regla simplemente coincide con una producción vacía, lo que significa que no hay ningún símbolo terminal o no terminal a la derecha de la producción. La producción simplemente asigna None al valor de p[0], lo que indica que no se ha ingresado ninguna expresión matemática y, por lo tanto, no hay nada que evaluar.

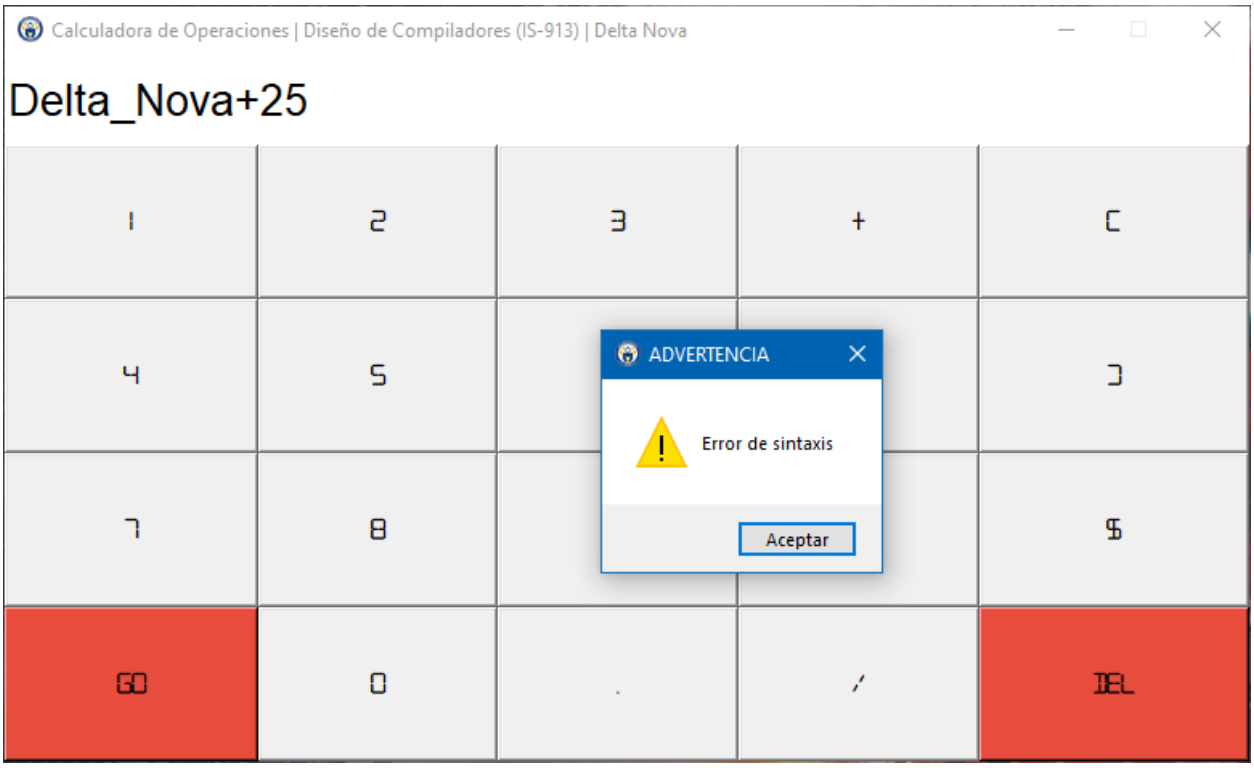
PROGRAMA EN EJECUCIÓN






Calculadora de Operaciones | Diseño de Compiladores (IS-913) | Delta Nova
Tokens:
LexToken(ENTERO,4,1,0)
LexToken(MAS,'+',1,1)
LexToken(ENTERO,2,1,2)
LexToken(POR,'*',1,3)
LexToken(PARENT_A,'(',1,4)
LexToken(ENTERO,5,1,5)
LexToken(MAS,'+',1,6)
LexToken(ENTERO,10,1,7)
LexToken(PARENT_C,')',1,9)
LexToken(MENOS,'-',1,10)
LexToken(ENTERO,27,1,11)
Operaciones: ('-', ('+', 4, ('*', 2, ('+', 5, 10))), 27)
Resultado: 7
© Derechos Reservados 2023 | Prohibida su copia, distribución parcial o total sin la autorización de Delta Nova


Calculadora de Operaciones | Diseño de Compiladores (IS-913) | Delta Nova
4+2*(5+10)-27Delta_Nova
ADVERTENCIA
¡Usted ha ingresado un caracter no permitido!
Aceptar


Calculadora de Operaciones | Diseño de Compiladores (IS-913) | Delta Nova
Delta_Nova+25
ADVERTENCIA
Error de sintaxis
Aceptar

HOJA DE DESEMPEÑO

Asignatura: Diseño de Compiladores (IS-913).
Sección: 0700.
Catedrático: Ing. Josian Cáceres.

		COMPETENCIAS A EVALUAR		
Nº Cuenta	**Nombre**	**Comunicación**	**Trabajo en Equipo**	**Resolución de Problemas**
20141011708	Kenet Orellana	100%	100%	100%
20161004203	Marvin Martínez	100%	100%	100%
20171031756	Luis Verde	100%	100%	100%

BIBLIOGRAFÍA

PLY (Python Lex-YACC) — PLY 4.0 documentation. (s. f.).

https://ply.readthedocs.io/en/latest/ply.html#lex-example

GeeksforGeeks. (2022). PLY Python Lex YaCC an Introduction. *GeeksforGeeks*.

https://www.geeksforgeeks.org/ply-python-lex-yacc-an-introduction/

Python 3 Tutorial. (s. f.). Tutorialspoint. https://www.tutorialspoint.com/python3/

Python 3 - GUI Programming (Tkinter). (s. f.). Tutorialspoint.

https://www.tutorialspoint.com/python3/python_gui_programming.htm