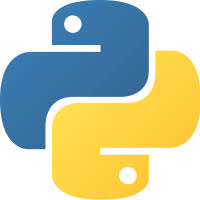


**LENGUAJES DE PROGRAMACION**

**PROYECTO:**

HERRAMIENTA EDUCATIVA PYTHON



**INTEGRANTES:**

DELGADO CAMPOVERDE PABLO

CELLY ASANZA JIMMY

SANCHEZ NARANJO EILEEN

**Introducción**

Un compilador es un programa que transforma las lineas de código que normalmente son instrucciones de alto nivel, en instrucciones legibles para la computadora. Para que un compilador pueda transformar dicho código a código maquina, se debe seguir una serie de pasos las cuales son: análisis léxico, análisis sintáctico, generación de código intermedio, optimización de código y genereación del código ensamblador. El presente trabajo tiene como objetivo de estudio, implementar los dos primeros pasos antes mencionados.

Para el análisis léxico de las sentencias escritas en un código fuente, se utilizará la herramienta **lex**. **Lex** es un programa que genera analizadores léxicos y ha sido muy usada a lo largo de los años con el auge de los sistemas Unix. A su vez, se utilizará el programa **yacc**, que por sus siglas en inglés significa Yet Another Compiler-Compiler, es usado junto con lex ya que es el encargado de generar el análisis sintáctico de las expresiones.

Esto quiere decir que **yacc** comprueba que la estructura organizada en el código fuente concuerde con las especificaciones sintacticas del lenguaje mediante **BNF**. Para el desarrollo de este proyecto se usará el lenguaje Python ya que es multiparadigma, lo cual permite varios estilos al momento de programar como la programación estructural, funcional, orientada a objeetos, etc.

Python posee de una librería muy util la cual permite hacer análisis léxico y sintáctico, el nombre de esta librería es **PLY** (Python Lex-Yacc) el cual implementa las herramientas de parseo Lex y Yacc.

**Objetivo**

Construir una interfaz gráfica que pueda visualizar el análisis y validación de la parte léxica (Lexer) de los tokens usados y visualizar la construcción de la parte sintáctica (Parser) con la librería PLY de python.

**Alcance**

El alcance del compilador se definirá para los condicionales **if** – **else** y funciones **(def)** definidas dentro de Python con paradigma estructural y funcional. Los casos utilizados serán los 10 ejercicios que se presentaran a continuación:

**Ejercicio 1:**

* Escriba un programa que pida la nota de un esetudiante y diga que si es mayor a 6 está aprobado y menor a 6 está reprobado.

**Ejercicio 2:**

* Escribir un programa que pregunte al usuario su edad y muestre por pantalla si es mayor de edad o no.

**Ejercicio 3:**

* Escribir un programa que determine el precio a pagar en una entrada del cine, el valor de los tickets es 5,50, sabiendo que:

-Los niños de 0-5 no pagan

-De 6 a 12 pagan la mitad

**Ejercicio 4:**

* Escriba un programa que determine si un número es par o impar.

**Ejercicio 5:**

* Los empleados de una fábrica trabajan en turno diurno. Se desea calcular el jornal diario de acuerdo con los siguientes puntos: "

-La tarifa de horas diurnas es de $40. "

-Caso de ser Domingo, la tarifa se incrementará en $100 en el turno diurno

**Ejercicio 6:**

* Determinar la cantidad de dinero que recibirá un trabajador por concepto de las horas extras trabajadas en una empresa, sabiendo que cuando las horas de trabajo exceden de 40, el resto se consideran horas extras y que estas se pagan al doble de una hora normal.

**Ejercicio 7:**

* Escriba una función que dado un radio calcule el area de un circulo, dado su radio.

**Ejercicio 8:**

* Escriba una función que dado un dividendo y un divisor devuelva su división pero si el divisor es cero retorne 'No se puede'

**Ejercicio 9:**

**Ejercicio 10:**

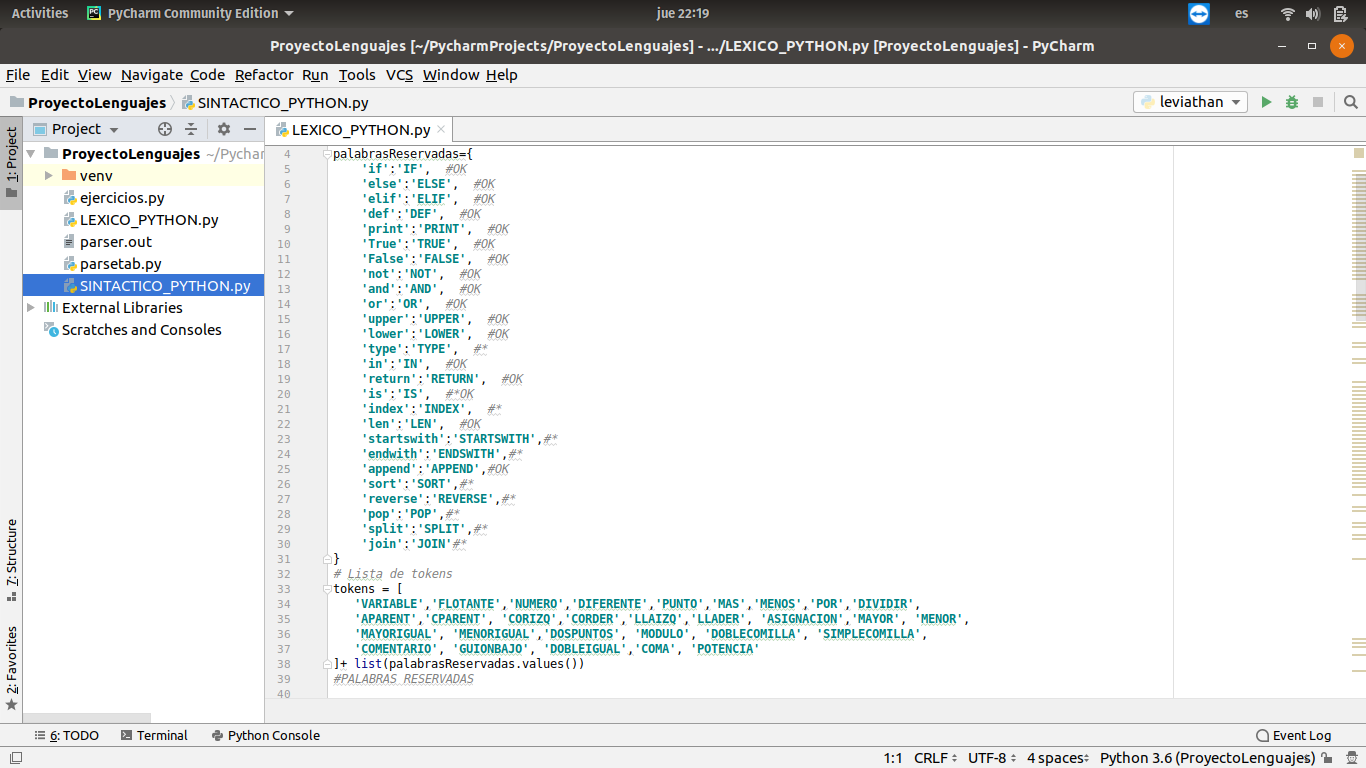
**Metodología**

La construcción del lexer validará los lexemas presentes en las instrucciones a evaluar por el mismo, estos también serán analizados a través de los patrones previamente definidos para los tokens. Estos patrones se definen a trvés de expresiones regulares, las cuales se encargan de limitar el número de lexemas posibles.

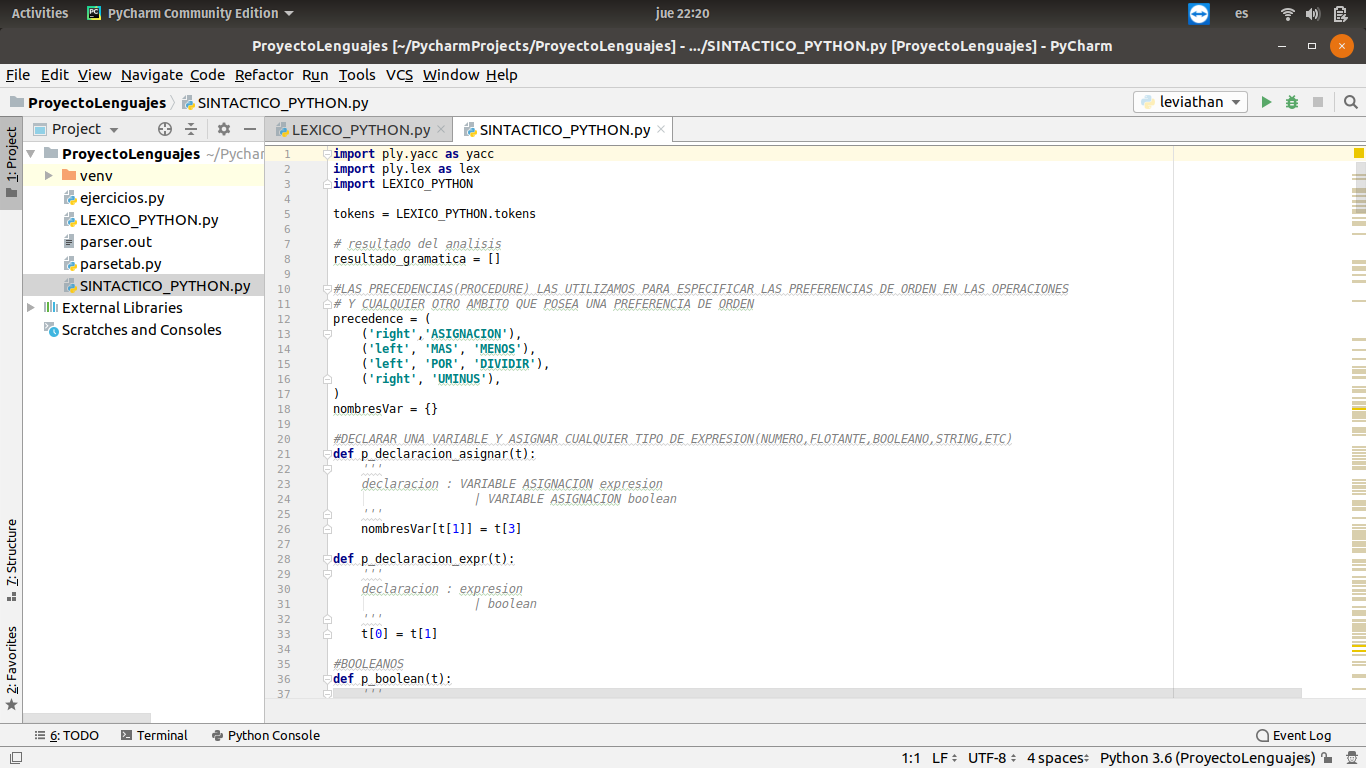
El objeto lex se encarga de realizar el análisis léxico, validando todos los lexemas en una cadena enviada de manera arbitraría, dicha cadena será válida si no existe algún lexema no válido en ella. Una vez que el conjunto de cadena es validado dentro del lenguaje, se procede a la construcción del árbol sintáctico a través del objeto yacc el cual se encarga en emular la parte sintáctica del lenguaje, esta parte se define a través de reglas BNF previamente creadas por los autores de este proyecto.

**Anexos**

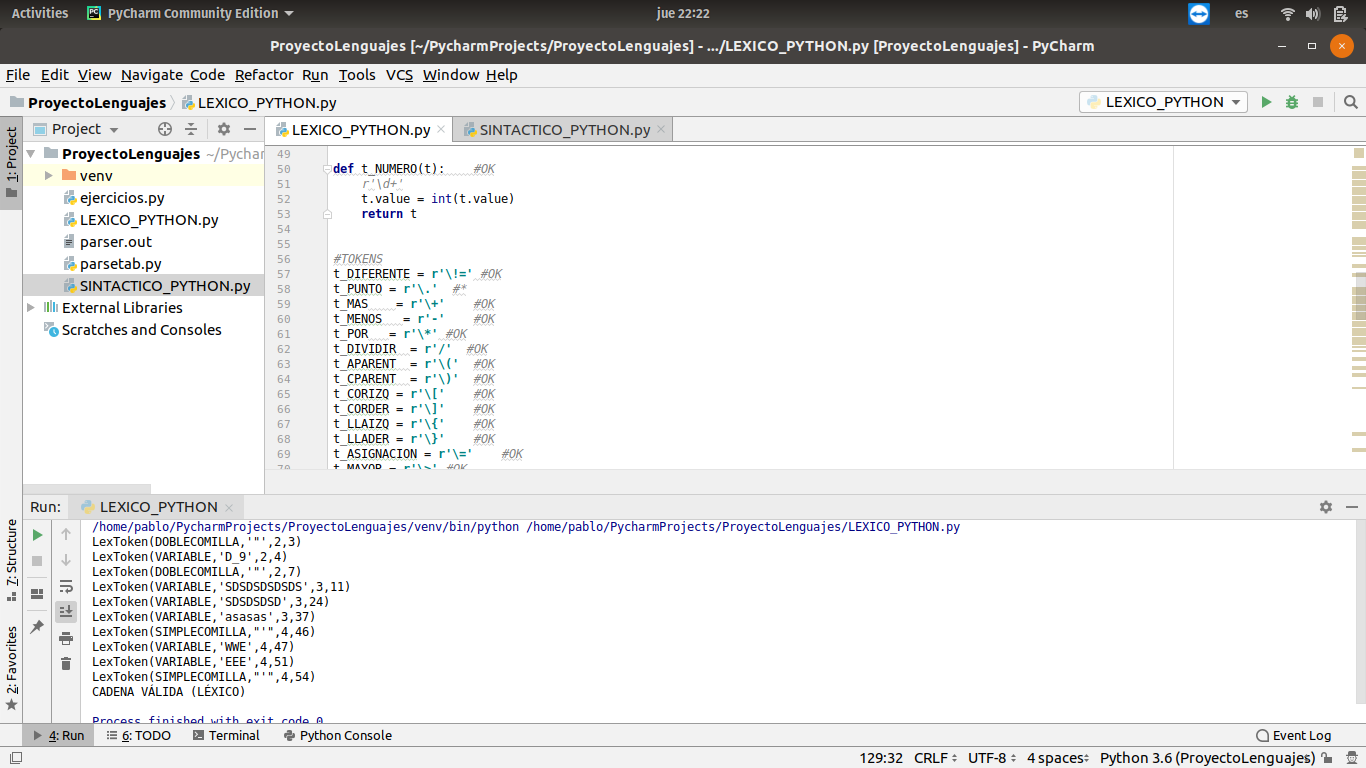
**LEXER**



**PARSER**

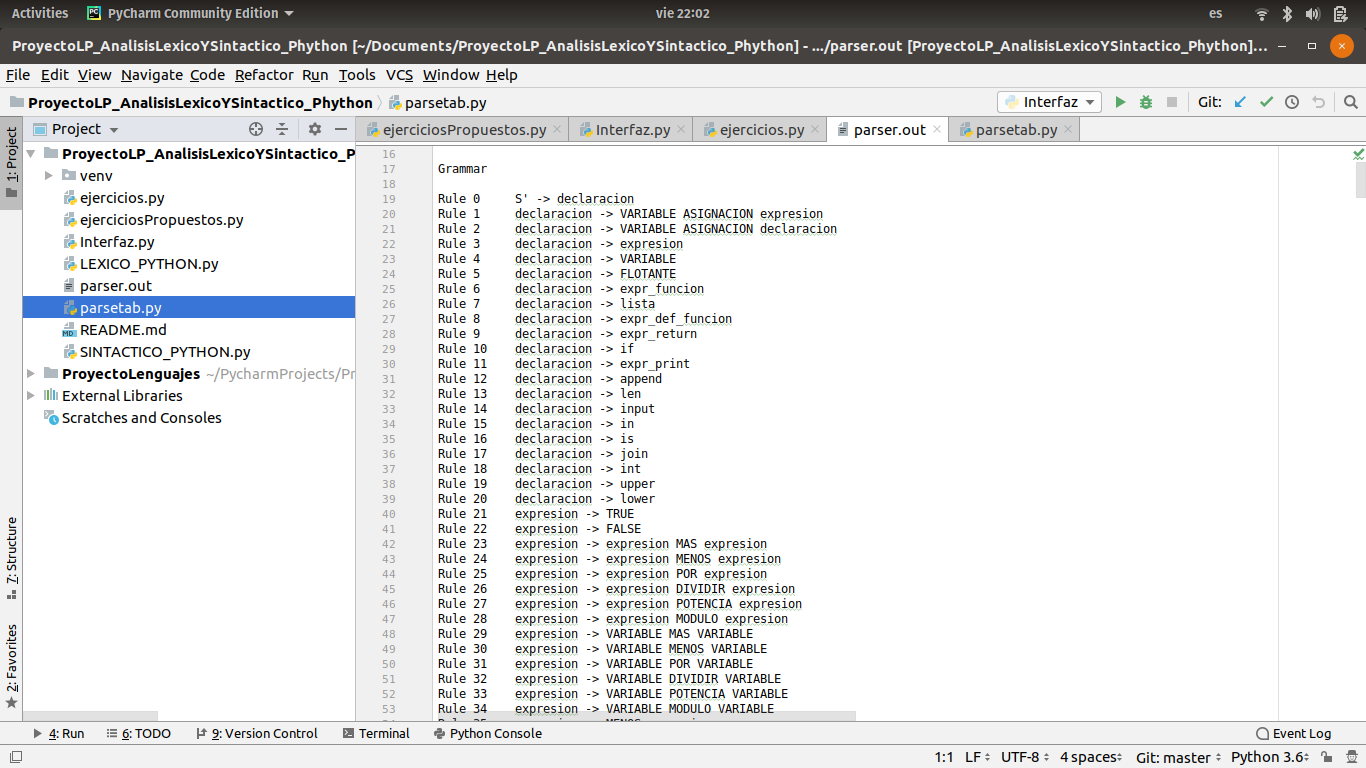


**EJECUCIÓN LEXER**



**EJECUCIÓN PARSER**

**PARSER.OUT**



**PARSERTAB.PY**

