Lilyac

Manual tecnico

Contenido

[Introducción 3](#_Toc21563062)

[Objetivo general del sistema. 4](#_Toc21563063)

[Objetivos específicos. 4](#_Toc21563064)

[Explicación del sistema 5](#_Toc21563065)

[Sistema 6](#_Toc21563066)

[Glosario 14](#_Toc21563067)

[Enlaces 15](#_Toc21563068)

[Apoyo técnico y asesoría 15](#_Toc21563069)

## Introducción

La finalidad de este manual técnico es proporcionar al lector la lógica con la cual se desarrolló la aplicación Lilyac, se describe detalladamente cada módulo del sistema así como su funcionamiento en conjunto todo esto para modificaciones futuras por programadores ajenos al proyecto.

## Objetivo general del sistema.

El objetivo es, básicamente es traducir un programa (o texto) escrito en un lenguaje fuente el cual llamaremos programa fuente, en un equivalente en otro lenguaje denominado objeto, al que llamaremos programa o código objeto.

Si el programa fuente es correcto en las tres etapas de su análisis (léxico, sintáctico, semántico), podrá hacerse su tal traducción; si no, se obtendrá un mensaje de error (o diversos) que permitirá ver claramente los posibles orígenes del error.

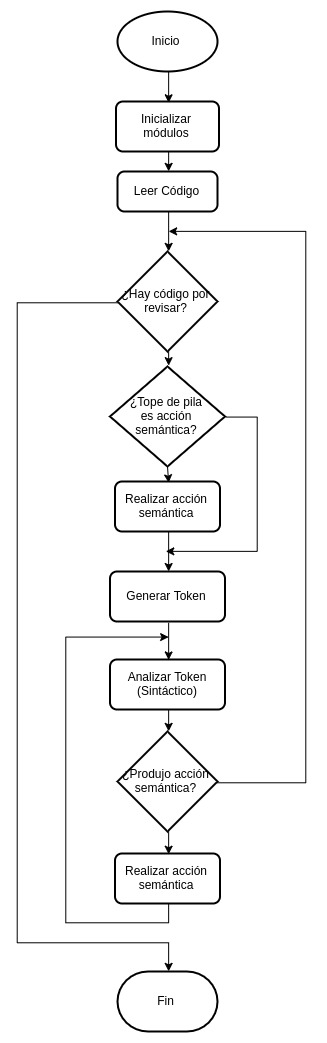
## Objetivos específicos.

1. Creación de software de base.
2. Capacidad de la creación de gramáticas.
3. Realizar un análisis léxico.
4. Realizar un análisis sintáctico.
5. Realizar un análisis semántico.
6. La creación de cuádruplos.

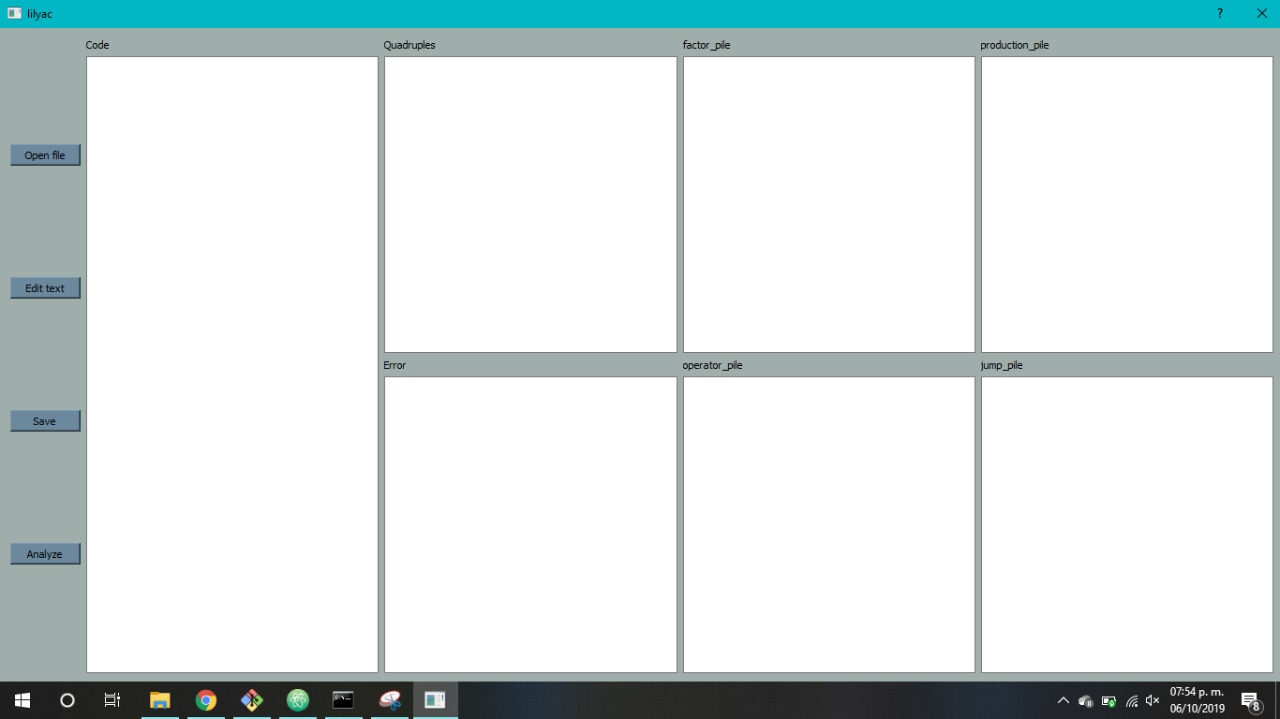
## Explicación del sistema

Al llamar al proceso de generación de código intermedio, primeramente se inicializan los 3 módulos involucrados (léxico, sintáctico y representación intermedia). Posteriormente, se lee el código fuente.

Mientras haya código por revisar; primeramente se revisa si el tope de la pila de símbolos del analizador sintáctico corresponde a una acción semántica, de ser así, se realiza está acción. De otro modo, se solicita al analizador léxico que produzca un token y este es pasado al analizador sintáctico. El analizador sintáctico realiza las derivaciones necesarias, si en algún momento durante el proceso una acción semántica es encontrada, esta es pasada al intermedio para su ejecución y el control se retorna al analizador sintáctico. Una vez que el analizador sintáctico termina el proceso de analizar un token, el ciclo se repite.



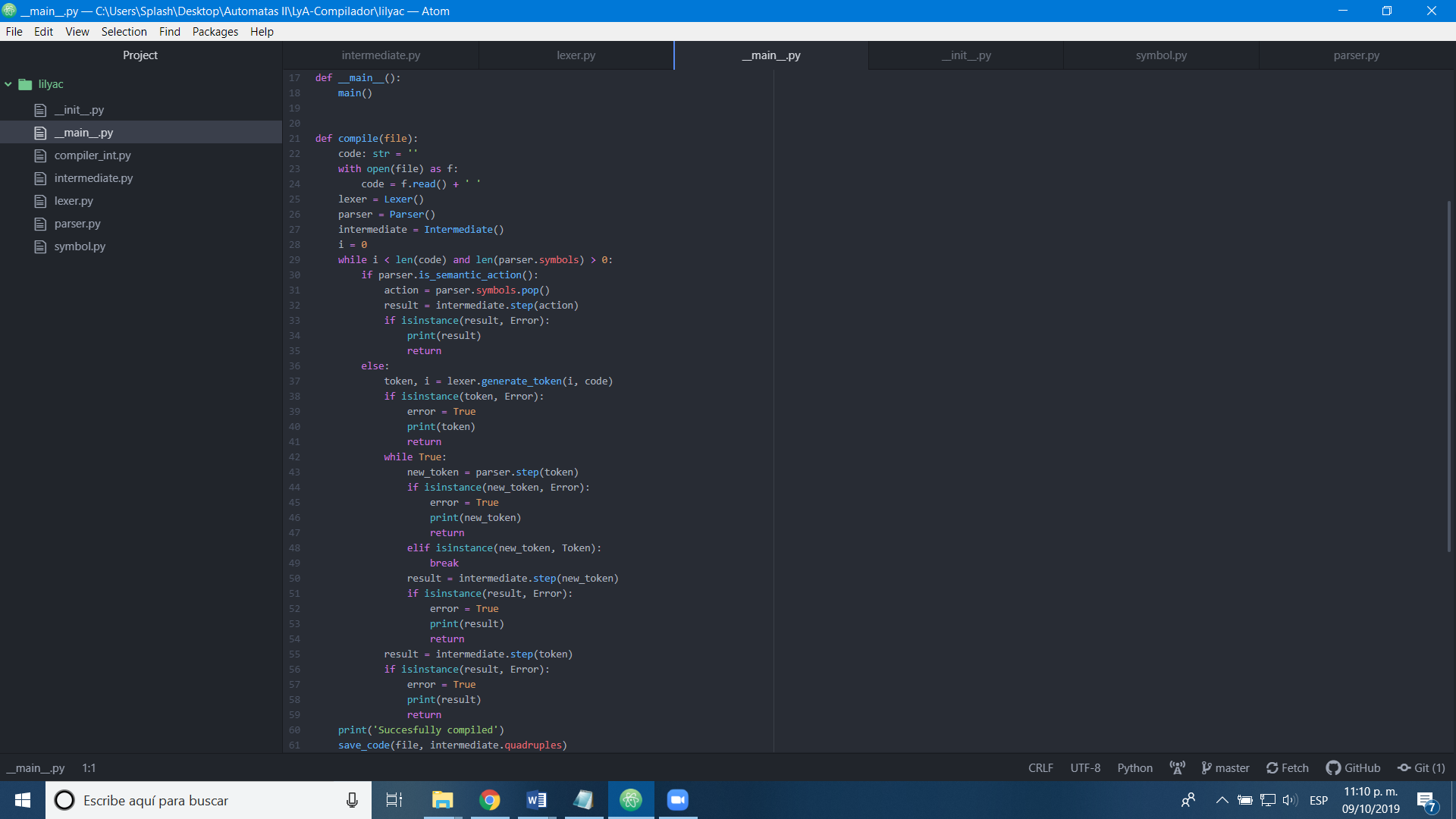
## Sistema



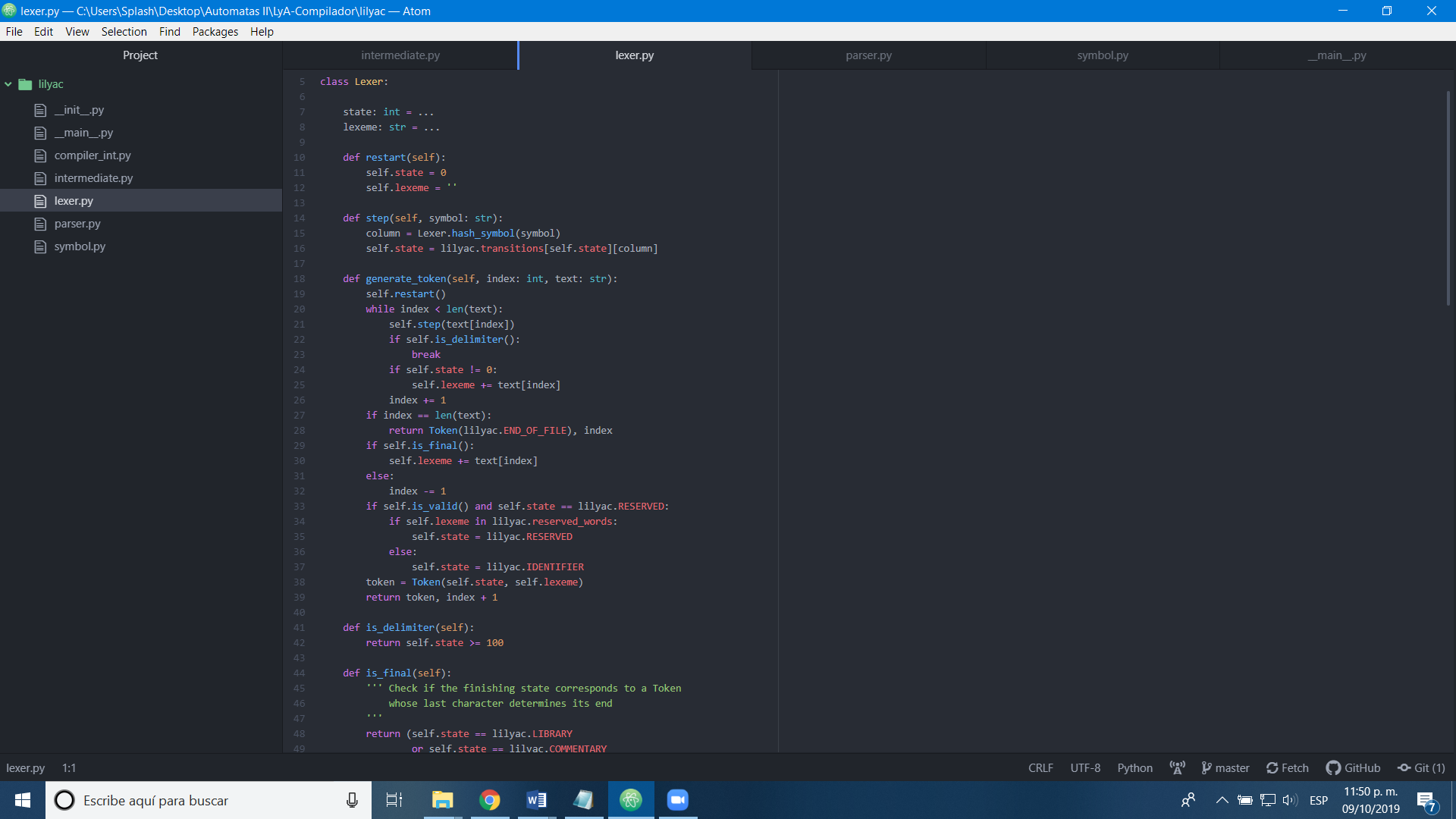
Dentro del botón analyze se encuentran en función el vaciado tanto de la tabla de cuádruplos, la pila y contador de cuádruplos, errores, pasos de la pila, pila tanto de operandos como operadores.

Al término de esta se procede a llamar a todos los métodos que constituyen la parte principal del programa y se imprimen en sus respectivos recuadros.

El código es el siguiente:

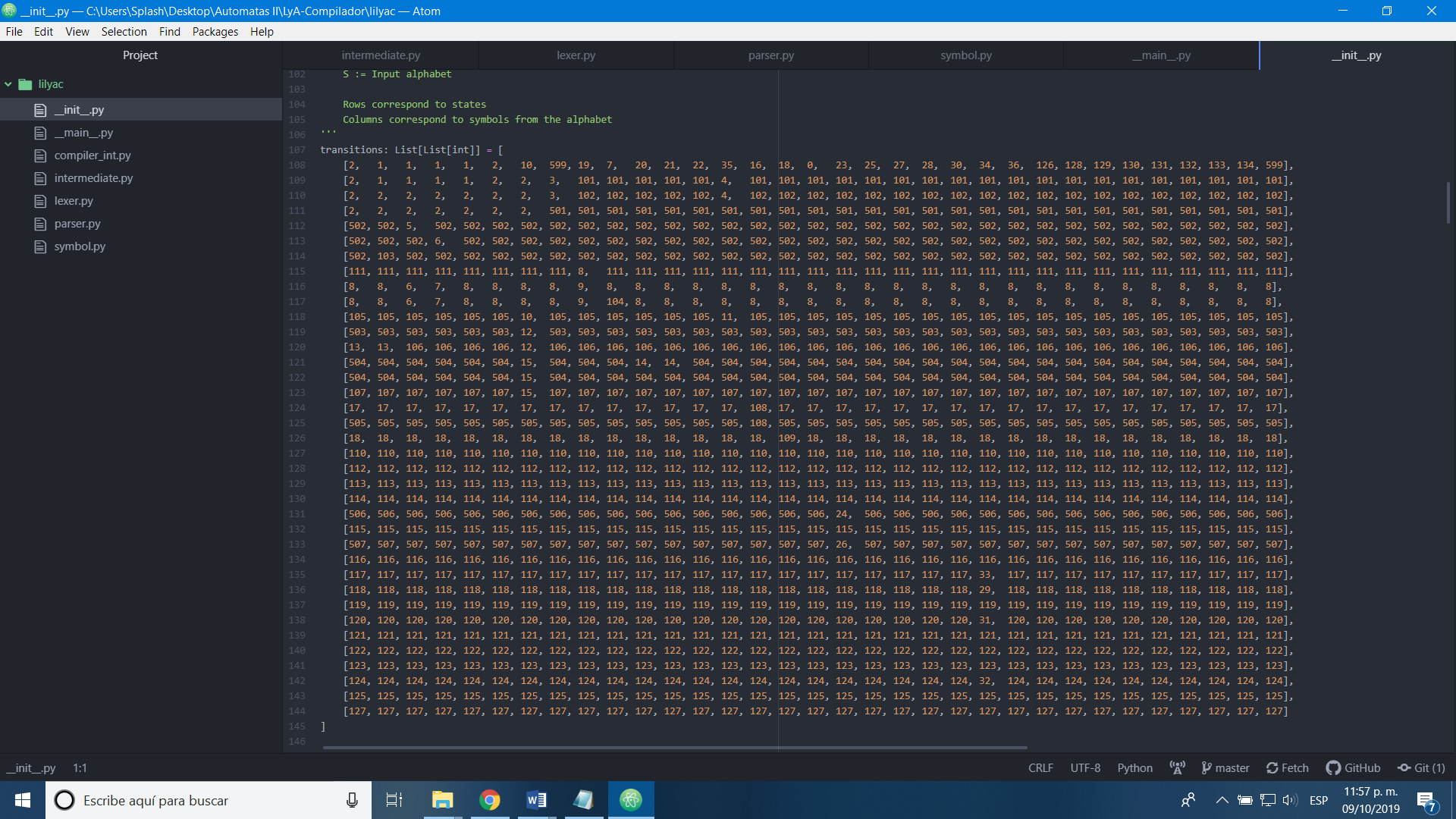
Dentro de este método se inicializan las clases de léxico, sintáctico y código intermedio.

Aquí de igual forma se ejecuta la acción de analizar mientras exista código dentro de la ventana de “code”.

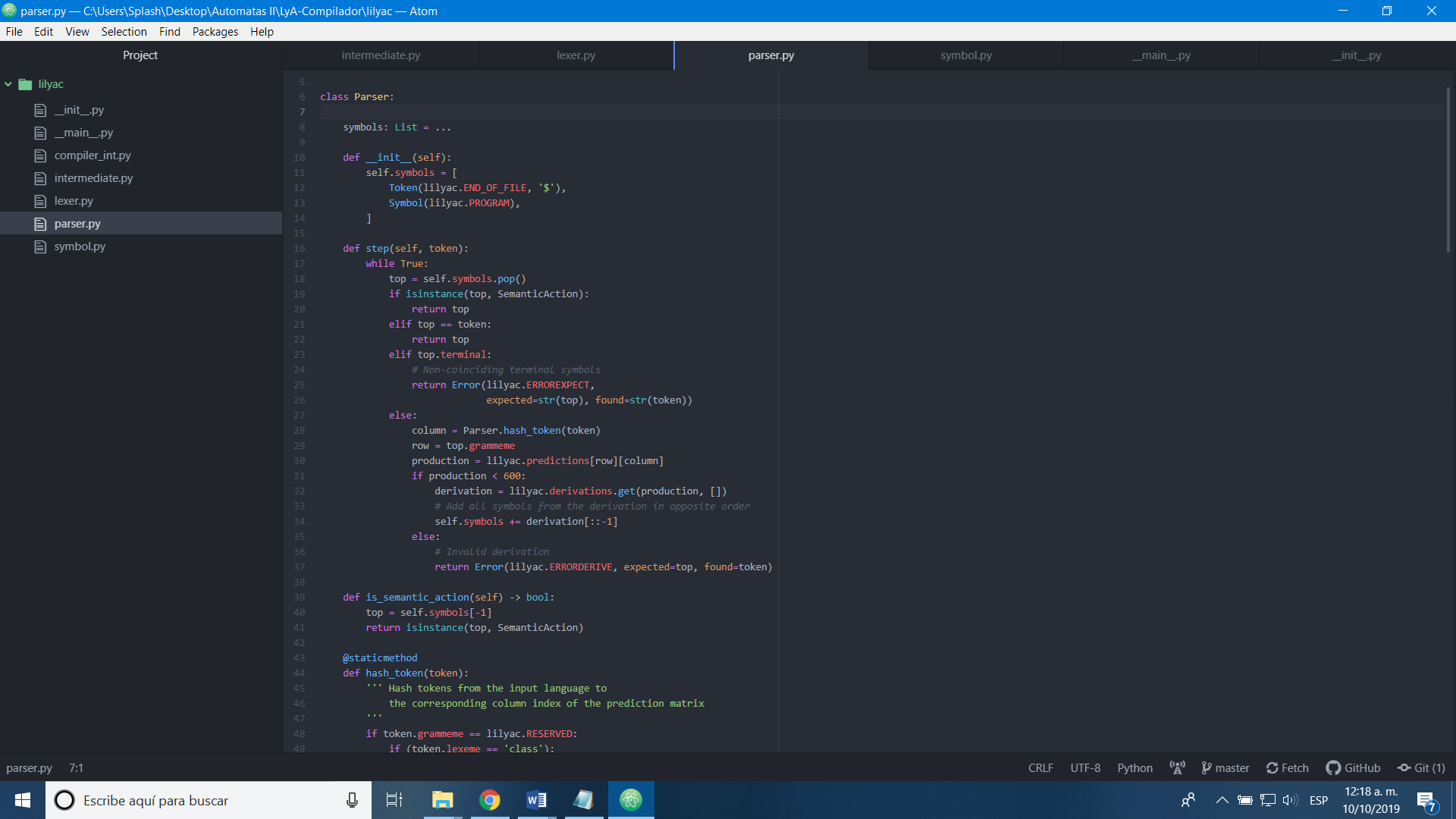
Dentro de la clase de léxico encontramos lo siguiente:

Aquí se realizan dos acciones las cuales son el inicio del índice para para las acciones y la otra es la identificación de tipos y generación de token.

Se indica cuando un estado es terminal o no (que puede ser seguido por otro carácter) y la relación con la matriz de predicción de léxico.

Matriz de predicción léxico:

Una vez analizado esto pasemos al sintáctico.

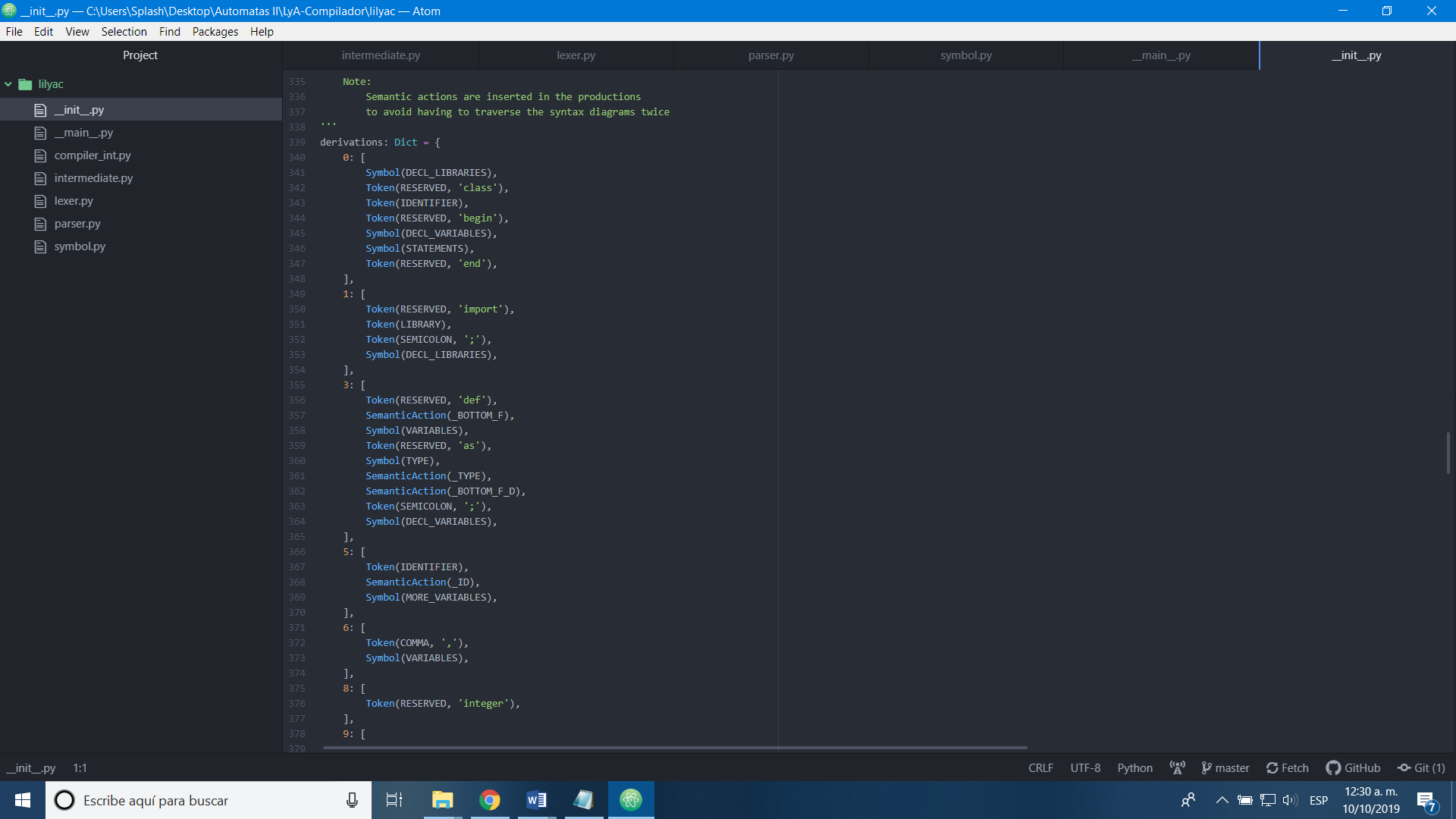


En esta sección del código se realiza la ejecución del análisis sintáctico, se utilizan stacks de acuerdo a la teoría de las pilas sintácticas.

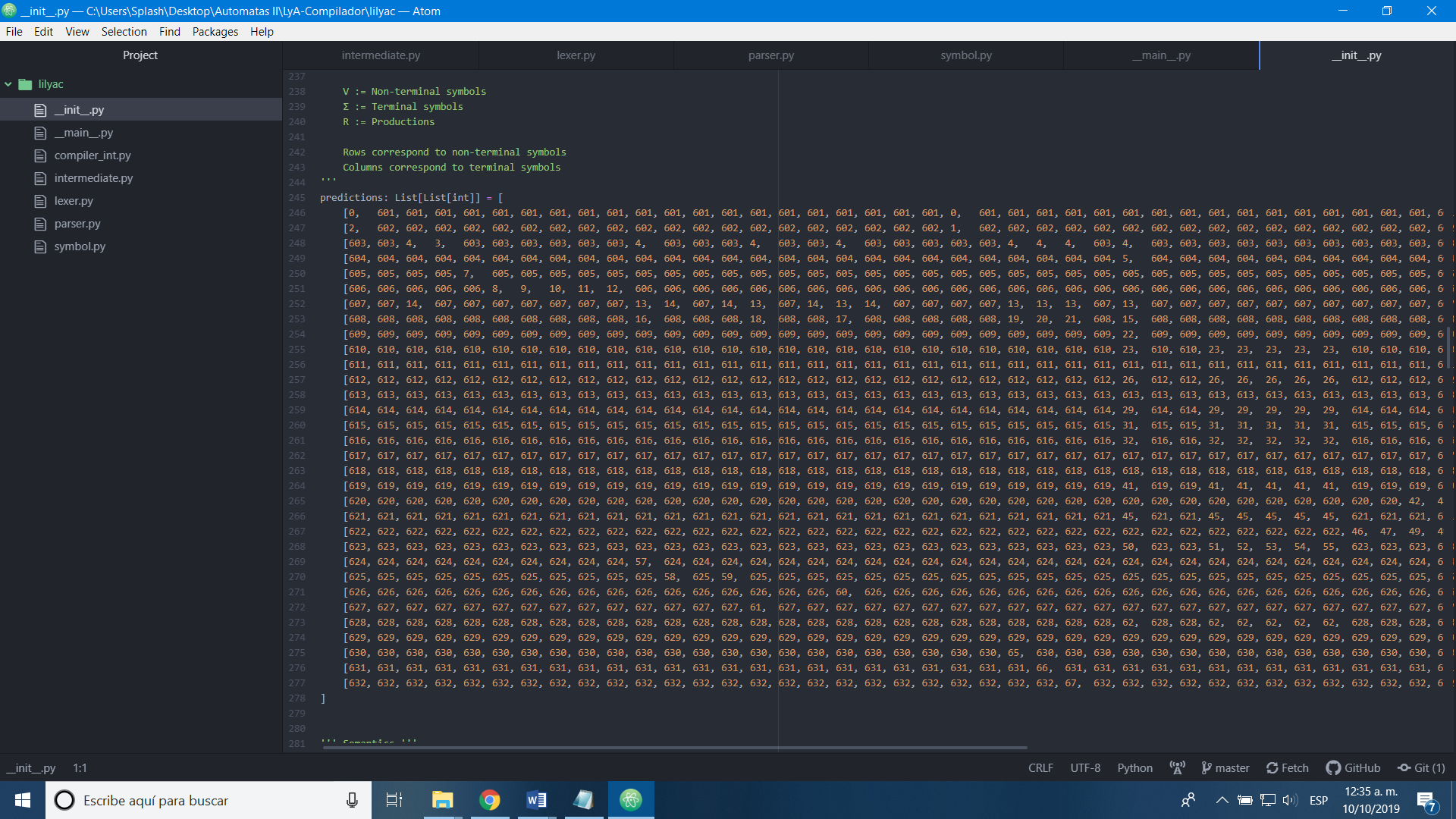
Se insertan los token que se sustraen del análisis léxico en la pila sintáctica y ya según las derivaciones que se mostrarán más adelante, se irán realizando las acciones.

De igual forma se hace uso de una matriz de predicción para realizar el análisis sintáctico.

Derivaciones:

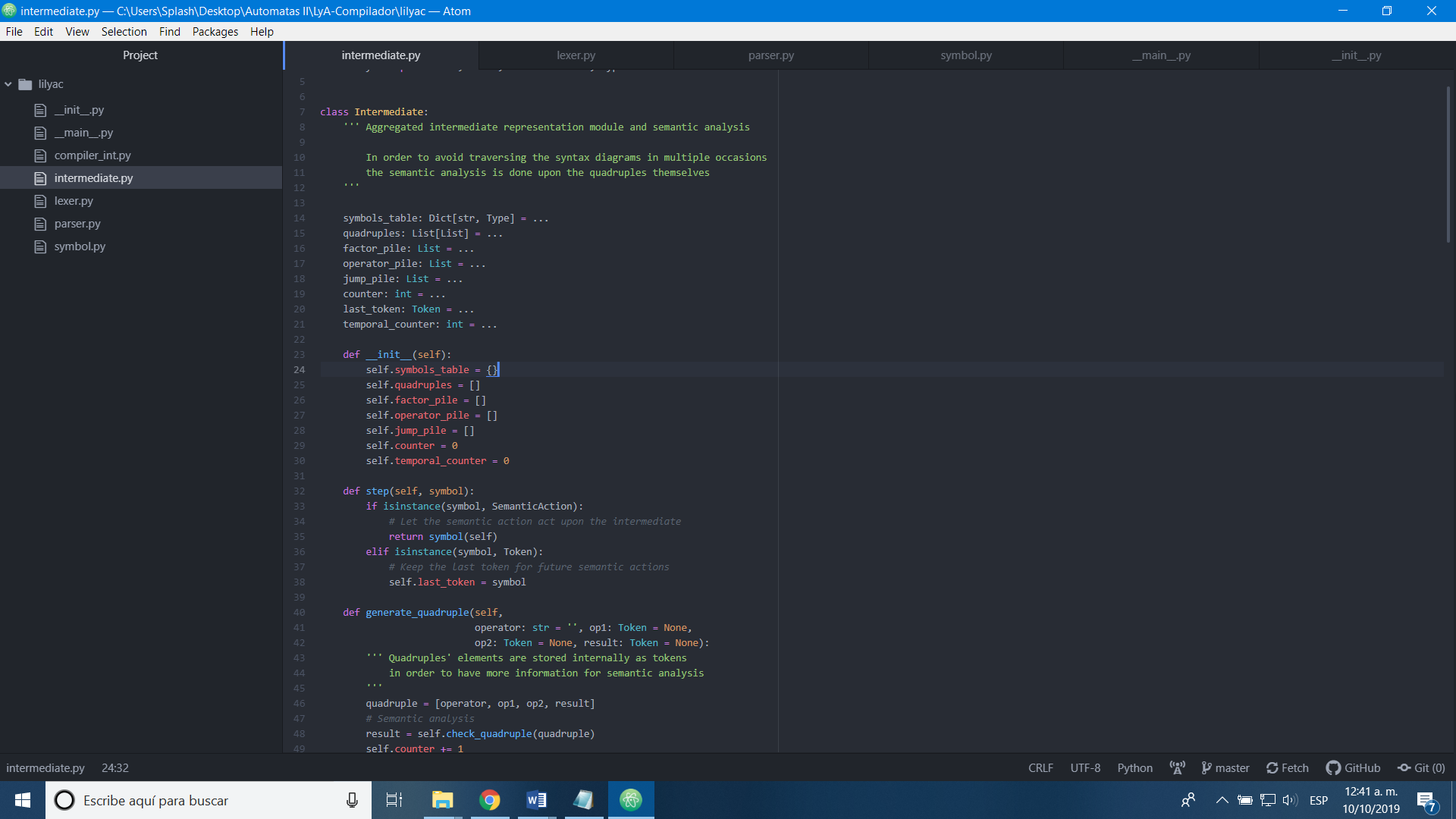


Matriz sintáctica

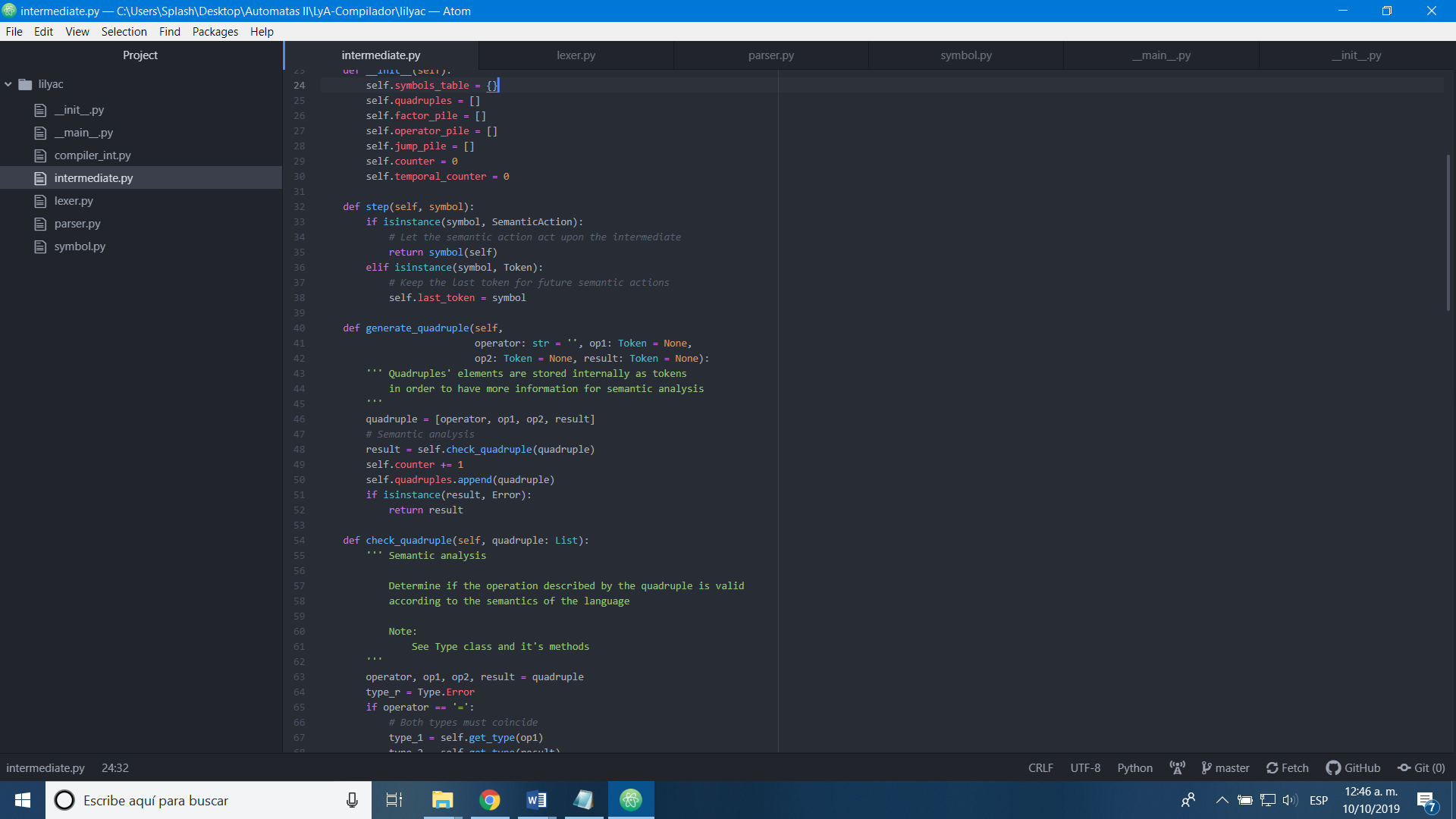


Análisis semántico y generación de cuádruplos:

Estás dos cosas se realizan en conjunto, normalmente se realiza el análisis léxico, sintáctico, semántico y por último la generación de cuádruplos, aquí se realizó de diferente manera, en primer lugar se realiza el cuádruplo y después se analiza semánticamente, todo esto para evitar pasar múltiples veces por el código.

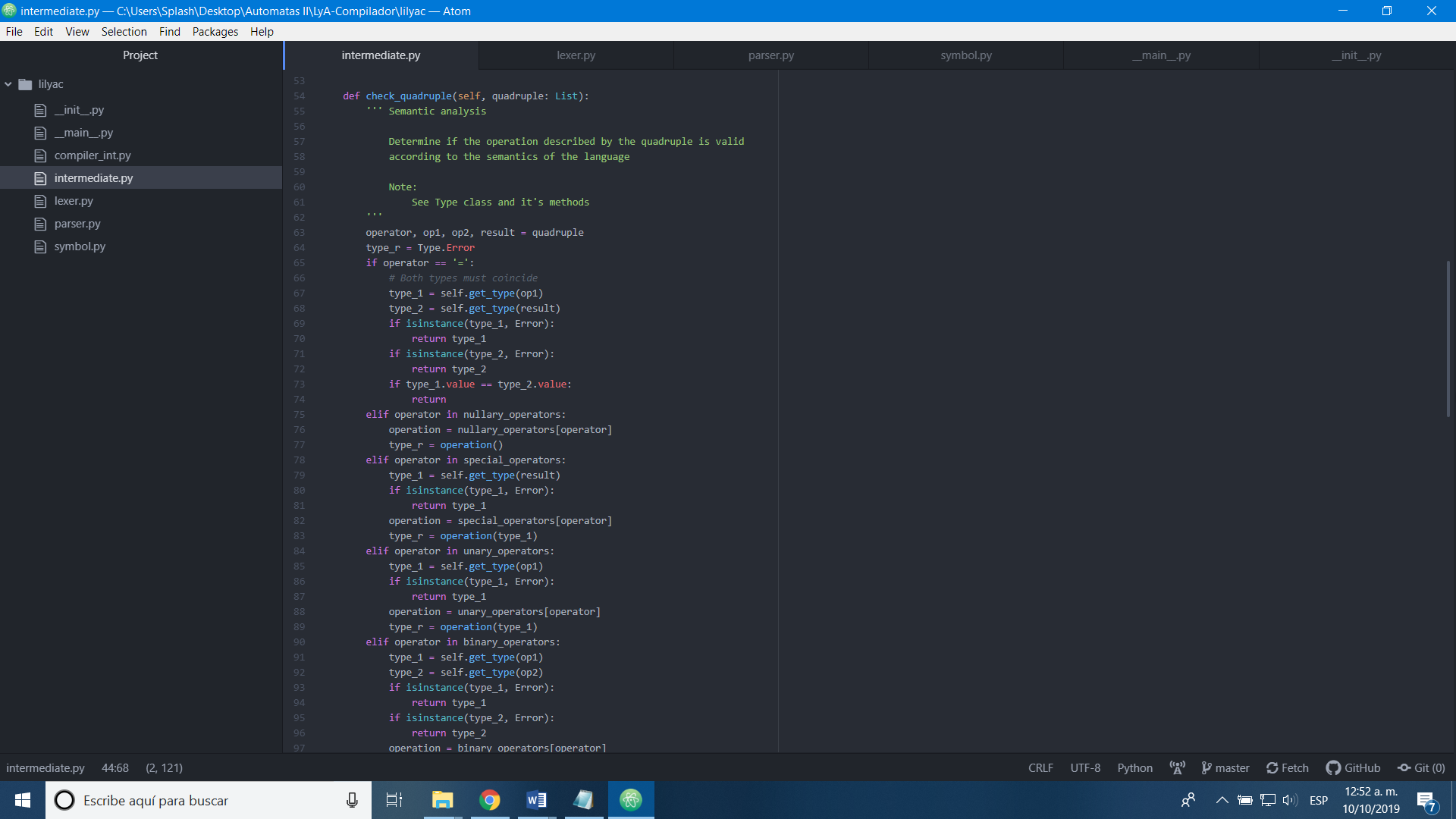
Aquí se inicializan las pilas de operandos, operadores, saltos y se genera la tabla de cuadruplos.

Se inicializa el contador temporal en 0 para comenzar el llenado de la tabla de cuádruplos.

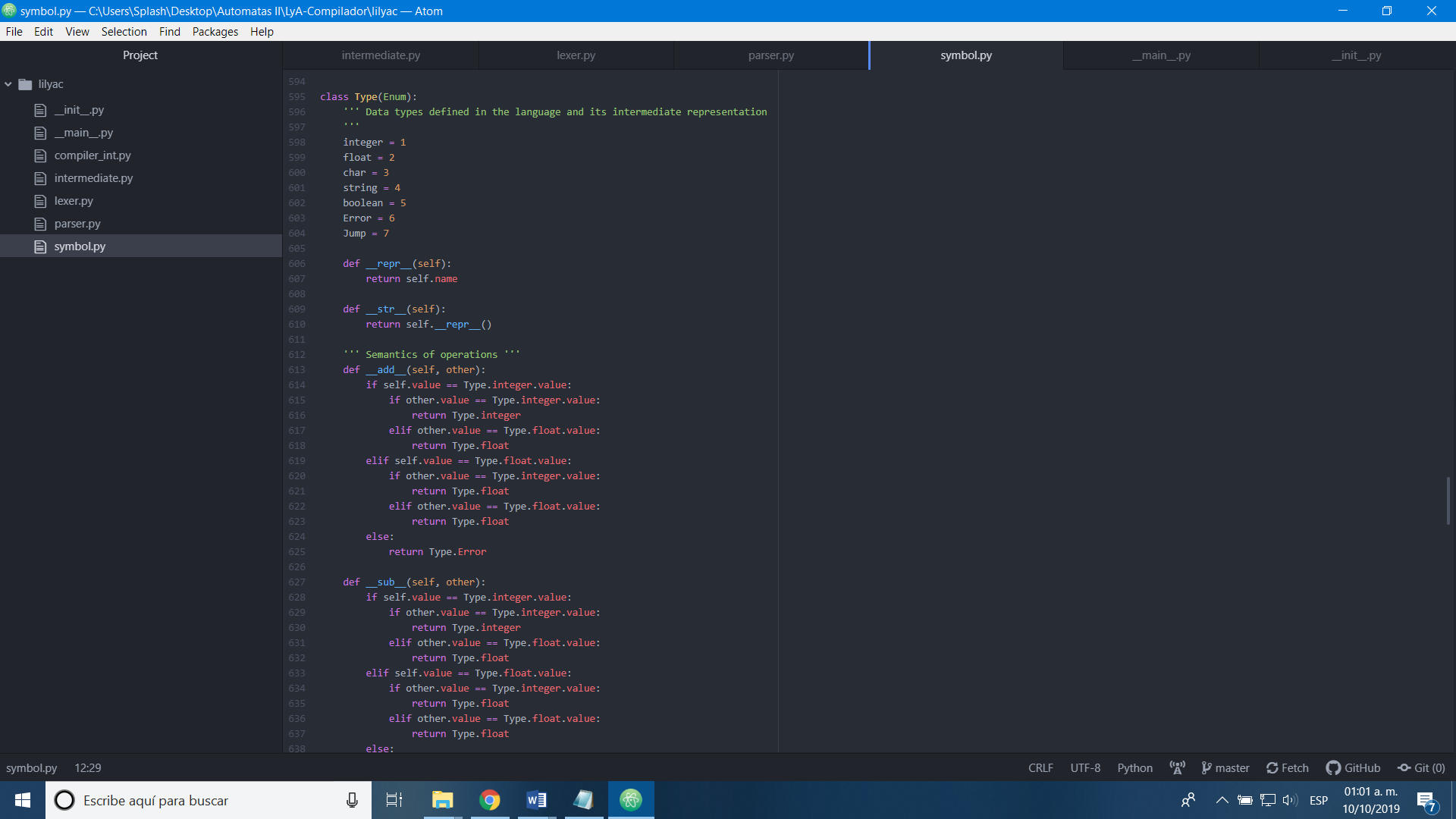
Generación de cuadruplo:

Los cuádruplos se almacenan internamente como tokens en orden para tener mas informacion para el análisis semantico.

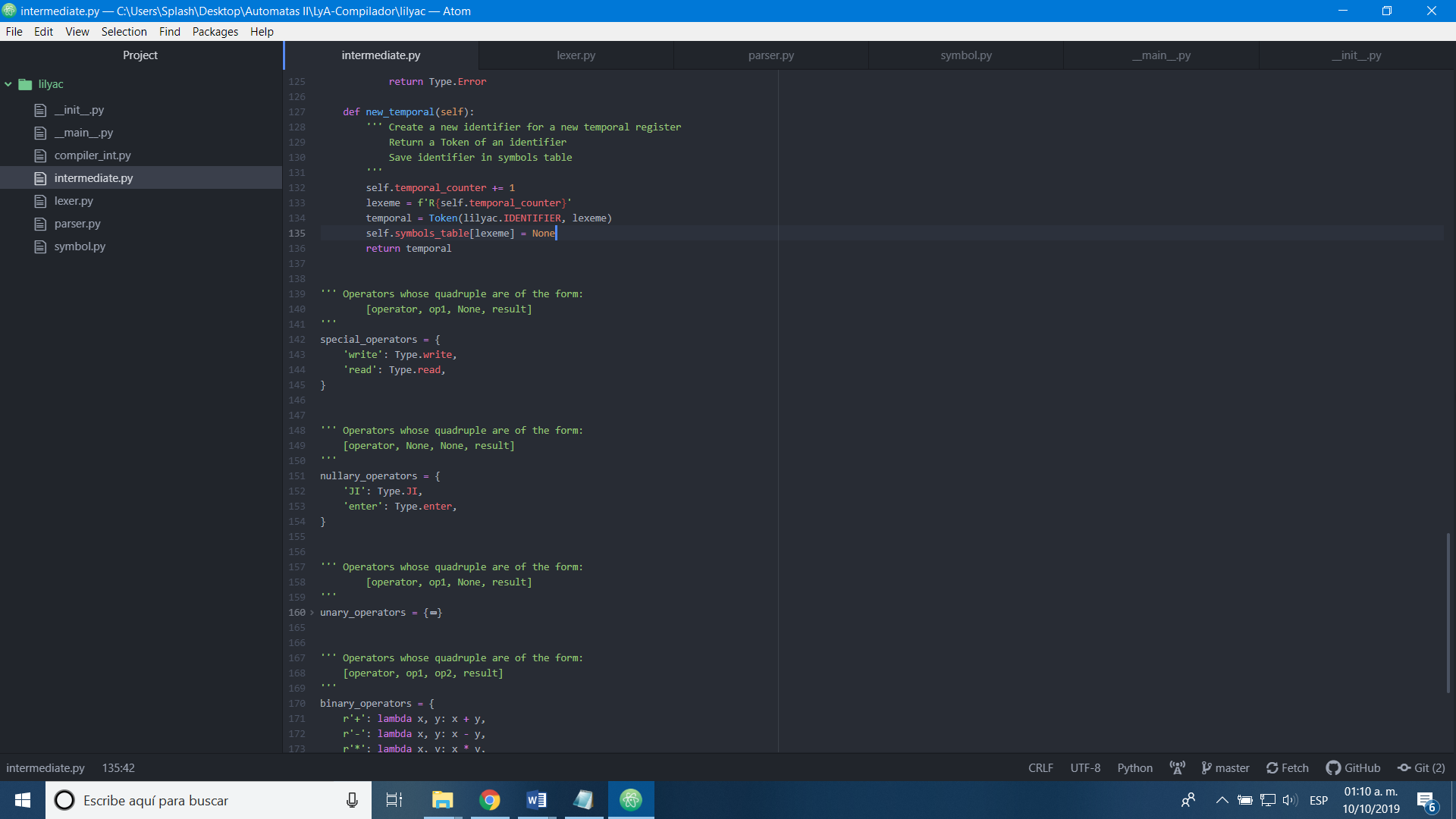
Análisis semántico:

Según los datos ingresados de los cuadruplos se evalua cada elemento, se retorna el valor del resultado obtenido y se determina si es correcto o es un error.

Para hacer el análisis entre tipo de operaciones no se realizó una matriz, se realizaron comparaciones que devolvían el valor que corresponde.

Operaciones semanticas:

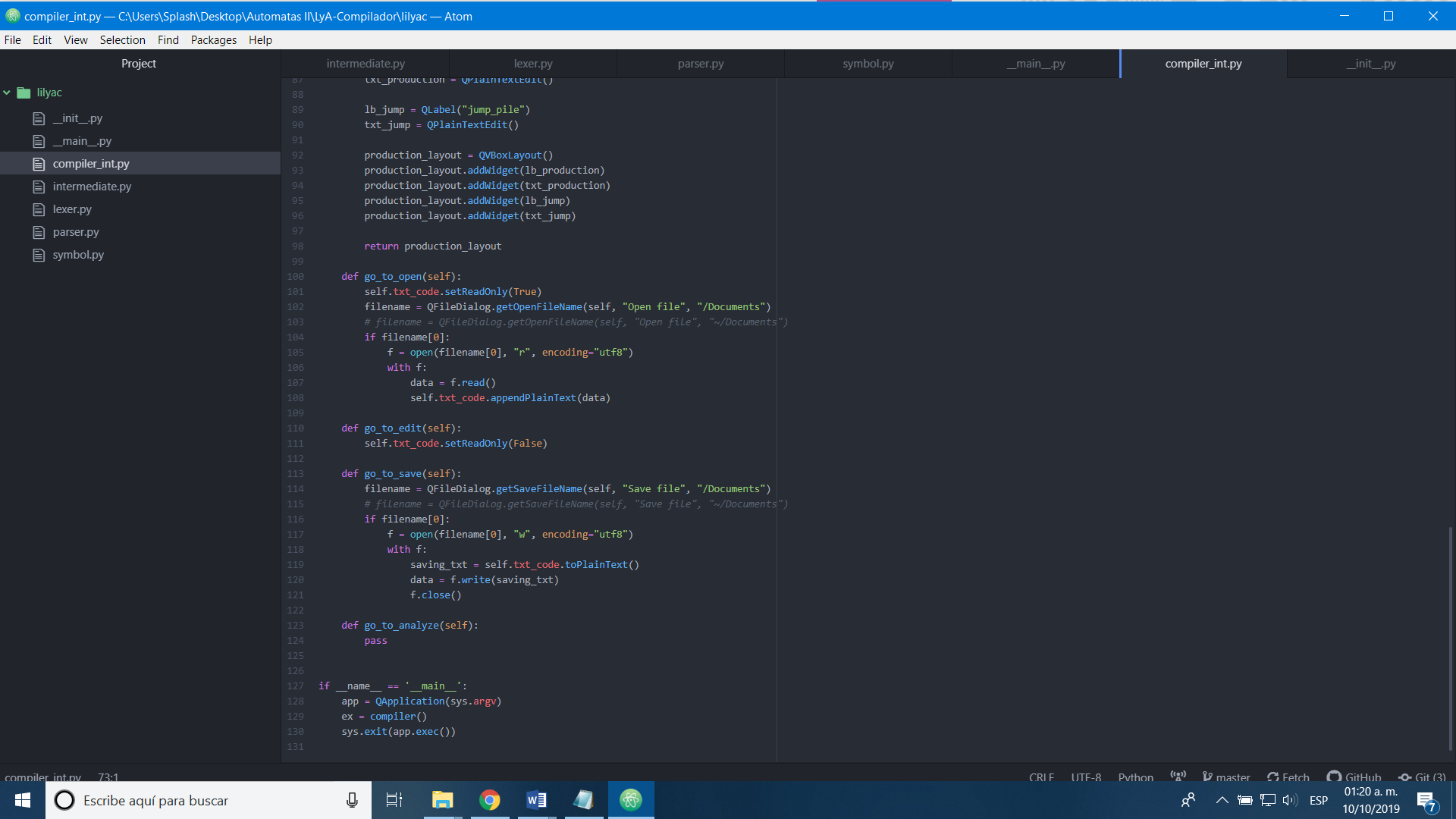
La representación de los tipos definidos en el lenguaje intermedio se asignó a valores diferentes para su mejor implementación.

Tipos de operadores extras:

Especiales: Write y read.

Nulos (No dependen de nada): Salto incondicional y enter.

Unario: salto en falso, salto en verdadero y “not”.

Open:

Edit:

Save:

Glosario

|  |  |
| --- | --- |
| Palabra | Definición |
| Léxico | Conjunto de palabras que constituyen una lengua. |
| Sintáctico | Es la parte de la gramática que estudia la forma en que se combinan y se relacionan las palabras para formar secuencias mayores como las oraciones. |
| Semántico | Ciencia lingüística que estudia el significado de las palabras y expresiones, es decir, lo que las palabras quieren decir cuando hablamos o escribimos. |
| Cuádruplo | Estructura de tipo registro con cuatro direcciones |
| Operando | Elemento sobre el cual se aplica una operación. |
| Operador | Símbolo utilizado para indicar la operación que se realiza entre los elementos que une o la relación que existe entre ellos. |

Enlaces

<https://github.com/CiSagelliv/lilyac>

Apoyo técnico y asesoría

Lilyac@compiler.com.mx