

**INSTITUTO TECNOLÓGICO DE MORELIA**

**División De Estudios Profesionales**

Departamento De Sistemas Y Computación

**LENGUAJES Y AUTÓMATAS II**

Unidad I: Análisis semántico.

**PRÁCTICA 2: ANALIZADOR LÉXICO DEL LENGUAJE PYTHON**

**PROFESOR**

BALLESTEROS LEÓN JAVIER

**ALUMNOS**

Ascencio Flores Ricardo - 15120269

Gil Jacome Arturo - 14121426

Martínez Aguilar Diego Ulises - 15121185

**MORELIA, MICHOACÁN, MÉXICO**

Jueves, 31 de enero del 2019

Tabla de contenido

[INTRODUCCIÓN 3](#_Toc536698504)

[**PRÁCTICA NO.1** 4](#_Toc536698505)

[CREAR UN ANALIZADOR LÉXICO PARA EL LENGUAJE DE PYTHON. 4](#_Toc536698506)

[DESARROLLO DE LA PRÁCTICA 4](#_Toc536698507)

[CONCLUSIONES 8](#_Toc536698508)

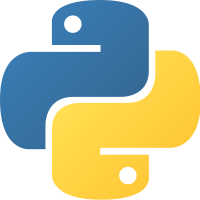
[REFERENCIAS 8](#_Toc536698509)

# INTRODUCCIÓN

**Python** es un lenguaje de programación interpretado cuya filosofía hace hincapié en una sintaxis que favorezca un código legible. Cuenta con estructuras de datos eficientes y de alto nivel y un enfoque simple pero efectivo a la programación orientada a objetos. La elegante sintaxis de Python y su tipado dinámico, junto con su naturaleza interpretada, hacen de éste un lenguaje ideal para scripting y desarrollo rápido de aplicaciones en diversas áreas y sobre la mayoría de las plataformas.

El intérprete de Python y la extensa biblioteca estándar están a libre disposición en forma binaria y de código fuente para las principales plataformas desde el sitio web de Python, *http://www.python.org/*, y puede distribuirse libremente. El mismo sitio contiene también distribuciones y enlaces de muchos módulos libres de Python de terceros, programas y herramientas, y documentación adicional.

El intérprete de Python puede extenderse fácilmente con nuevas funcionalidades y tipos de datos implementados en C o C++ (u otros lenguajes accesibles desde C). Python también puede usarse como un lenguaje de extensiones para aplicaciones personalizables.



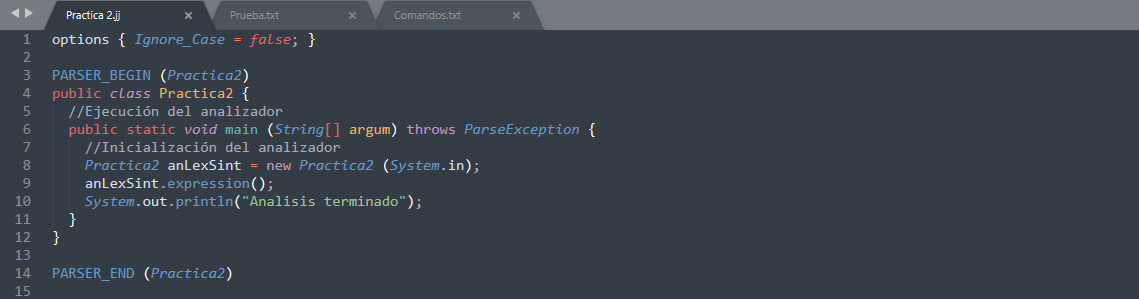
**Figura I1:** Logo Python.

**Modo interactivo**: el intérprete de Python estándar incluye un modo interactivo en el cual se escriben las instrucciones en una especie de intérprete de comandos: las expresiones pueden ser introducidas una a una, pudiendo verse el resultado de su evaluación inmediatamente, lo que da la posibilidad de probar porciones de código en el modo interactivo antes de integrarlo como parte de un programa. Esto resulta útil tanto para las personas que se están familiarizando con el lenguaje como para los programadores más avanzados.

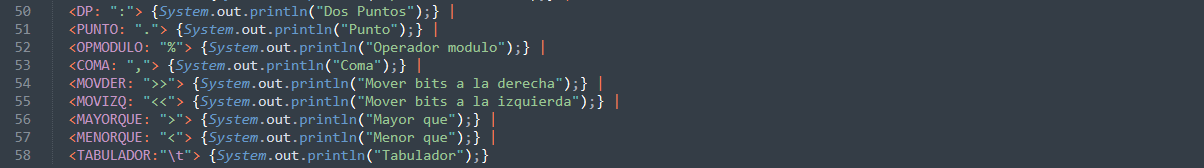
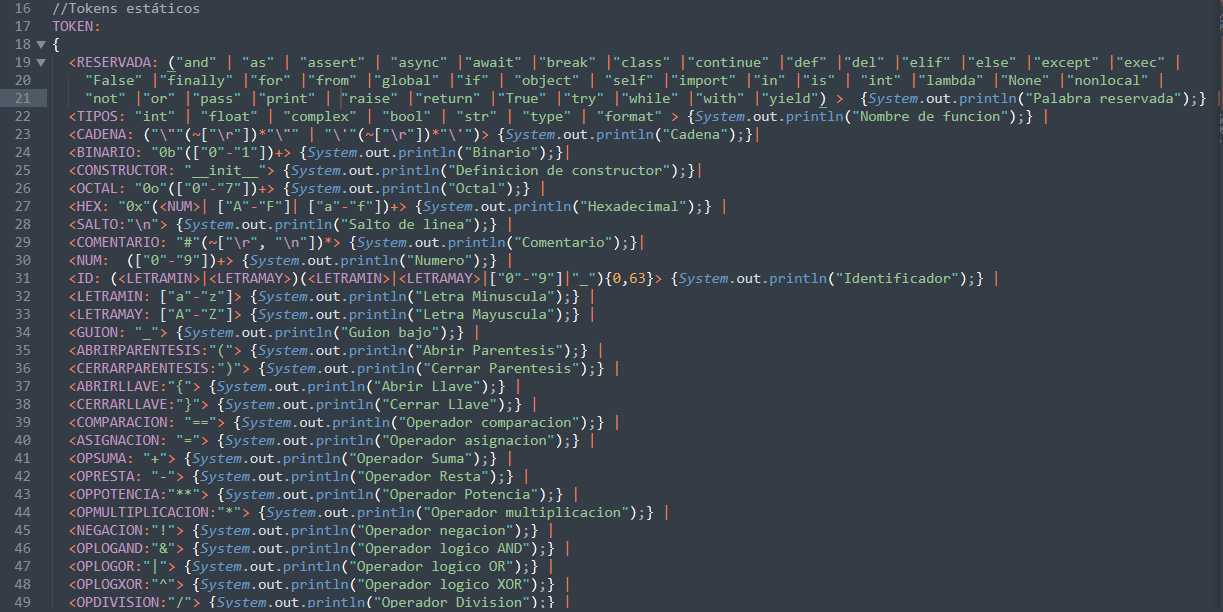
# **PRÁCTICA NO.1**

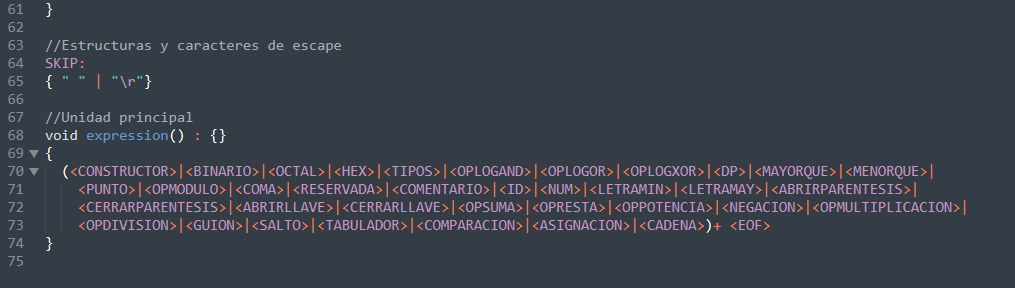
## CREAR UN ANALIZADOR LÉXICO PARA EL LENGUAJE DE PYTHON.

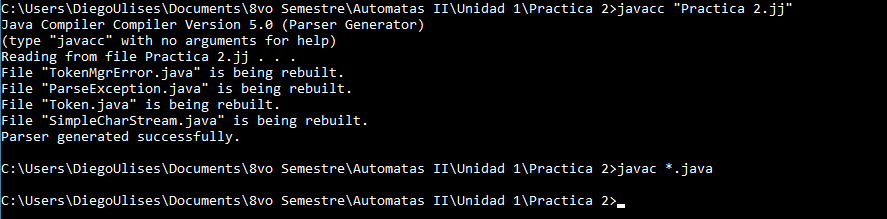
### DESARROLLO DE LA PRÁCTICA

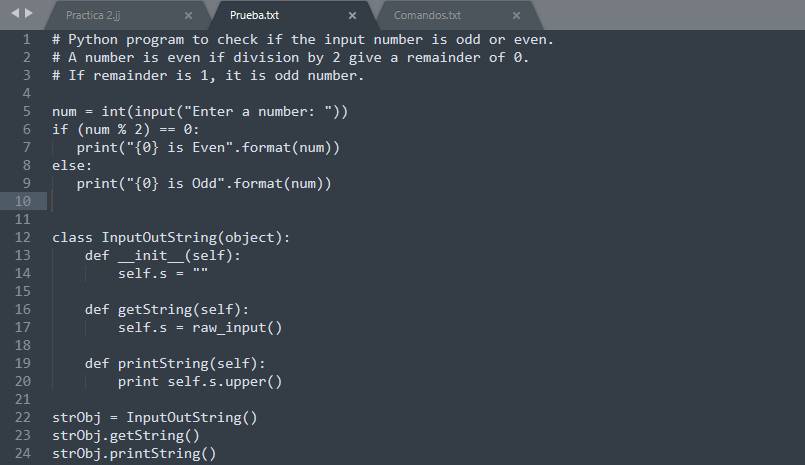


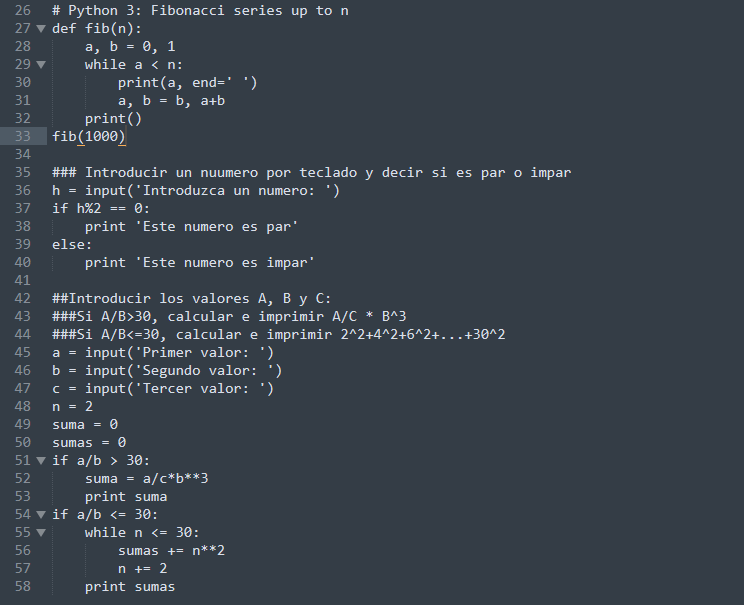
**Figura 1:** Se crea un archivo nuevo con extensión **.jj** y se establece la estructura necesaria para la ejecución e inicialización del analizador.

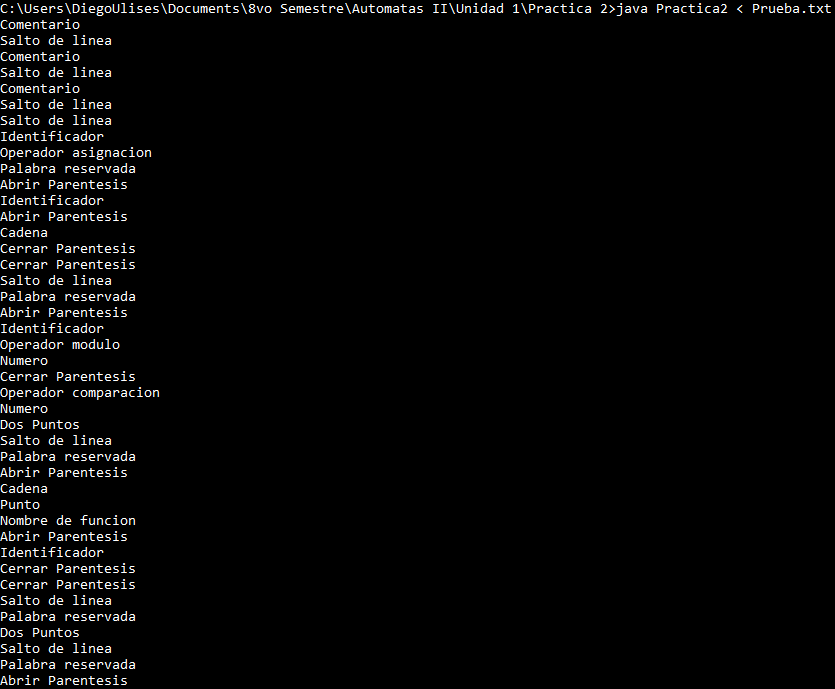
**Figura 2:** Se establecen los tokens estáticos que nos permitirán reconocer los los lexemas establecidos.

** Figura 3:** Se establecen las estructuras de escape, así como también se indica la unidad principal como aquella que va a aceptar la gramática establecida.

** Figura 4:** Para compilar este fichero, se debe hacer con "javacc" y posteriormente con "javac".



** Figura 5:** En un archivo de prueba se crea con extensión **.txt** el cual será evaluado por el analizador léxico antes creado, tal archivo contiene ejemplos en desorden de cada uno de los lexemas.

** Figura 6:** Para ejecutar el programa y dentro del símbolo del sistema con los comandos de **java Practica1 < prueba.txt** con el cual comprobamos el funcionamiento del analizador léxico, no trata de comprobar las estructuras del lenguaje ni compilará el fichero en código binario.

# CONCLUSIONES

El analizador léxico es una parte fundamental de un compilador, ya que se dedica a verificar que todas las palabras y símbolos pertenecen al lenguaje cuyo código se está tratando de compilar. En el caso de Python hay ciertas características que lo hacen diferente a la gran mayoría de lenguajes de programación, por ejemplo, la carencia del **;** además de que no hay necesidad de declarar los tipos de datos de las variables a utilizar, por lo que el análisis léxico para éste se vuelve de cierta forma mas simple al tener menos elementos a reconocer.

# REFERENCIAS

*[1] https://www.python.org/*

*[2] https://www.comoprogramar.org/ejemplos-de-python-codigo/*

*[3] http://www.pythondiario.com/p/ejercicios-de-programacion-python.html*