**INSTITUTO POLITÉCNICO NACIONAL**

**ESCUELA SUPERIOR DE CÓMPUTO**

**COMPILADORES**

Tecla Parra Roberto

Saldaña Aguilar Gabriela

González Orta Edwin

3CM3

MANUAL TÉCNICO

Logo

**INTRODUCCIÓN**

Nuestro objetivo es enseñar los principios de programación de una manera divertida y fácil haciendo que la programación sea una competencia accesible para todos los niños en el mundo.En el futuro (muy cercano) todo lo que haremos requerirá habilidades básicas de programación y, por lo tanto, es importante aprender esta habilidad y aprender a gustar

Hacer programación visual proporciona premios muy rápidos para los esfuerzos, por lo que es perfecto para los niños pequeños que a menudo tienen problemas para establecer metas a largo plazo. Logo cuenta con varias versiones.

Papert desarrolló un enfoque basado en su experiencia con Piaget a principios de los sesenta. Fundamentalmente consiste en presentar a los niños retos intelectuales que puedan ser resueltos mediante el desarrollo de programas en LogoEs conocido por poder manejar con facilidad gráficas tortuga, listas, archivos y recursividad.

Logo es uno de los pocos lenguajes de programación con instrucciones en español en algunos intérpretes, entre ellos: FMSLogo, LogoWriter, WinLogo, Logo Gráfico, XLogo, MSWLogo y LogoEs.

En nuestro trabajo no emularemos el trabajo completo de logo si no lo más básico como son: Variables, Ciclos, Condiciones, Funciones y Recursividad. Queremos mostrar el diseño de un compilador haciendo uso de este proyecto tipo logo.

**REQUERIMIENTOS MÍNIMOS DE SISTEMA**

Procesador Intel Core 2 duo

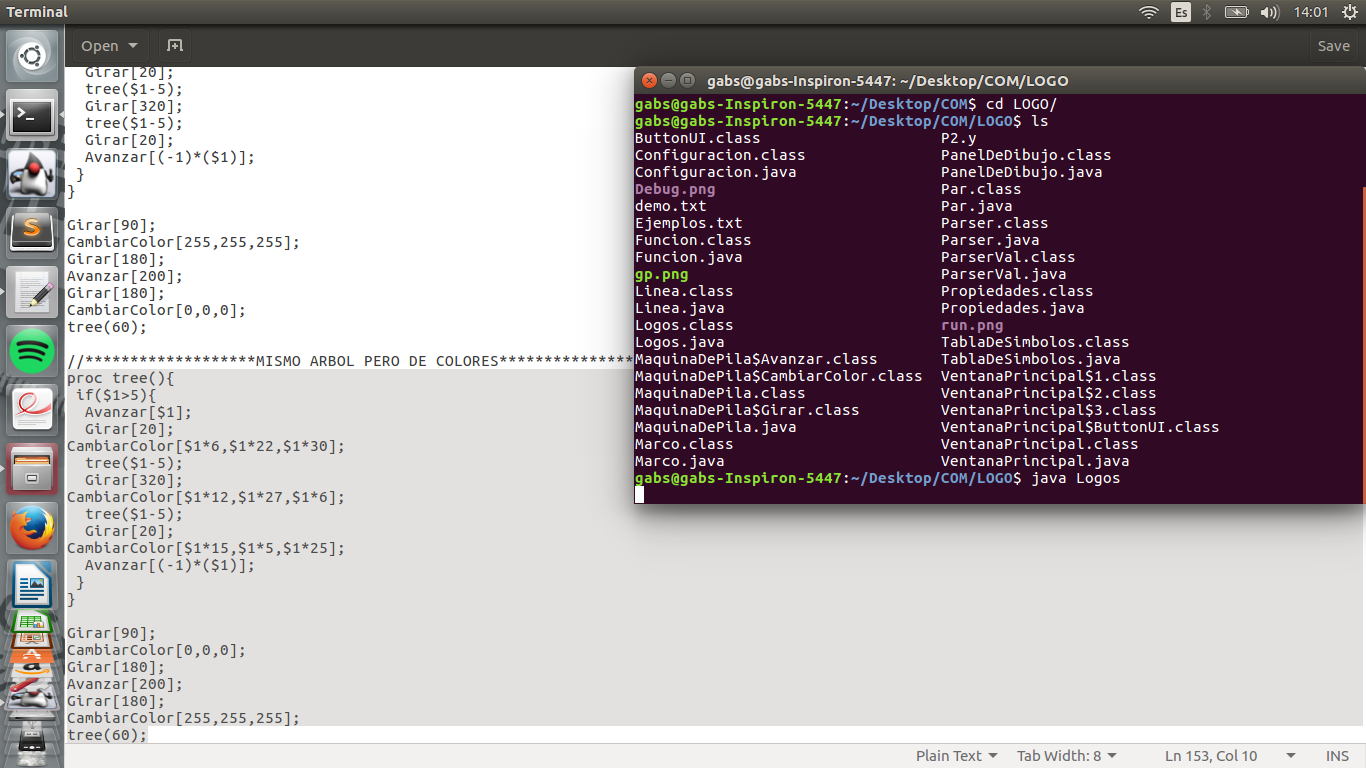
Al menos 2 gb en memoria RAM

Al menos 200 GB en memoria ROM

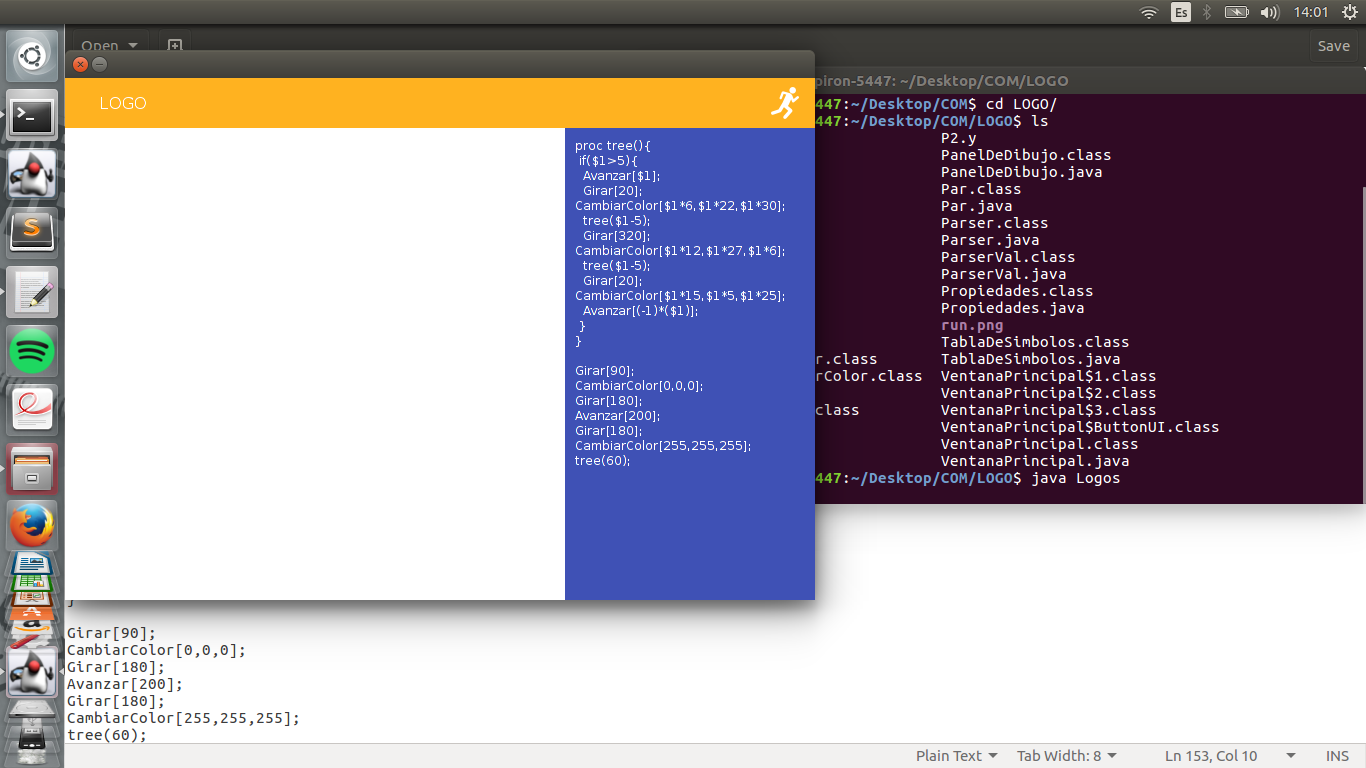
Sistema Operativo con Java instalado (jdk, jre)

**Instalación y Ejecución**

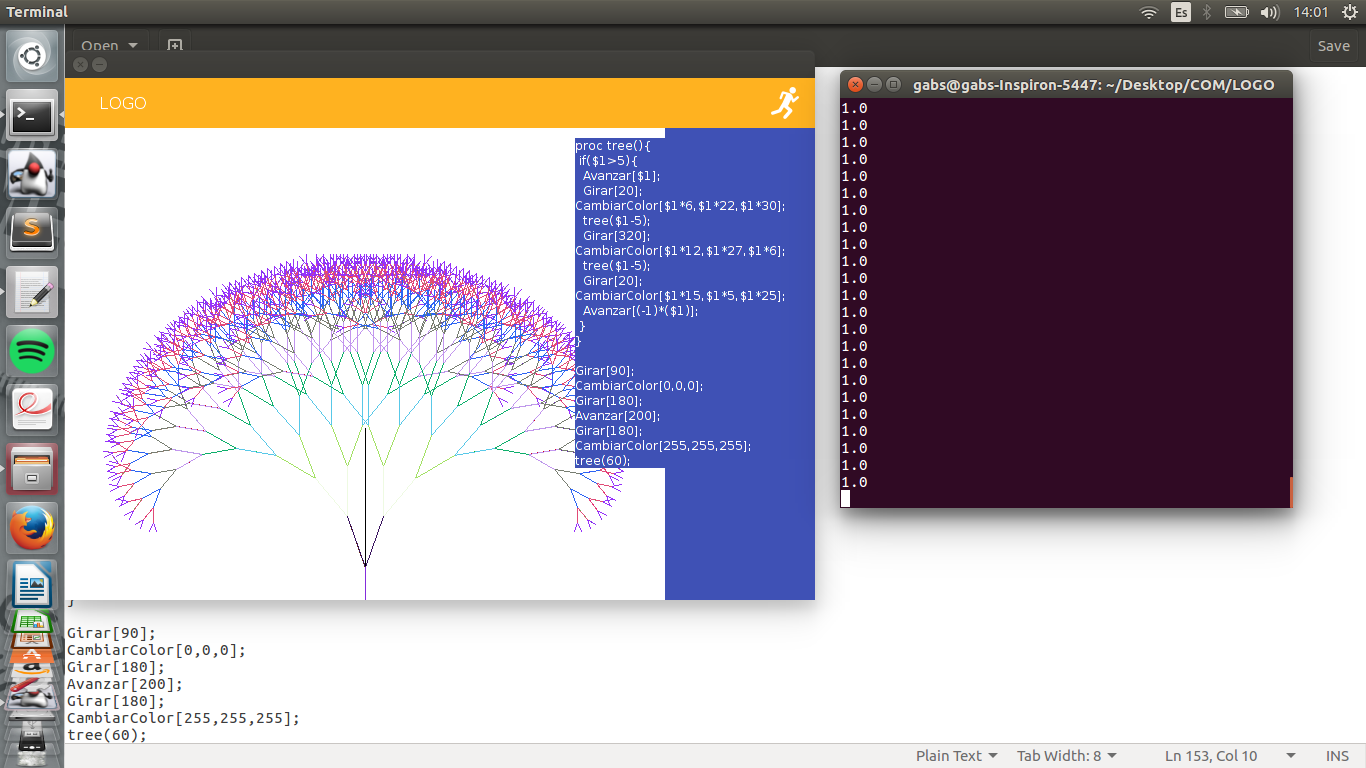
1. Una vez instalado Java, procedemos a movernos a la carpeta donde guardamos el proyecto con la terminal: cd Desktop/COM/Logo



1. Una vez estando dentro de la carpeta del archivo ejecutar el siguiente comando en la terminal: java logos



1. Una vez que el proyecto este ejecutando podremos hacer uso de funciones y comandos nativos de logo, en este caso copiaremos y copiamos del archivo de ejemplos:

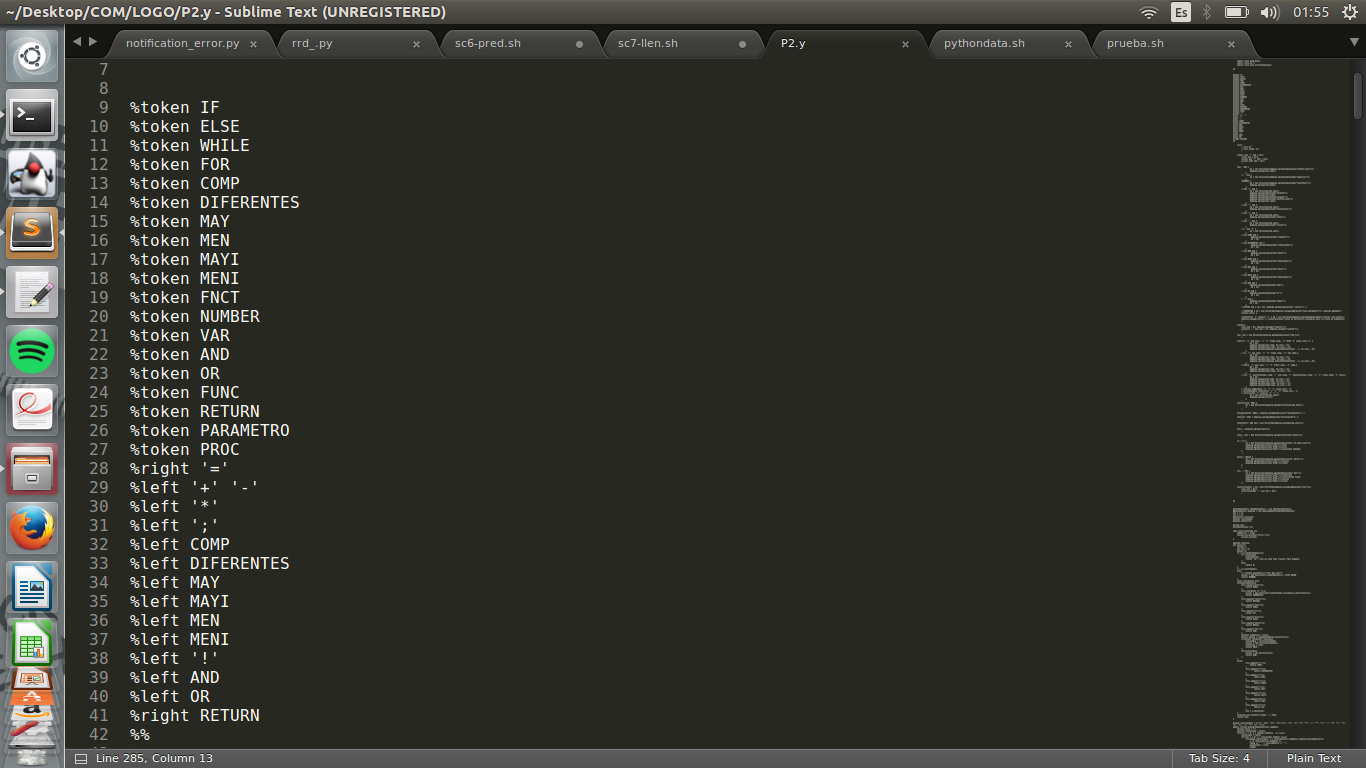


Las funciones predefinidas que pueden emplearse son:

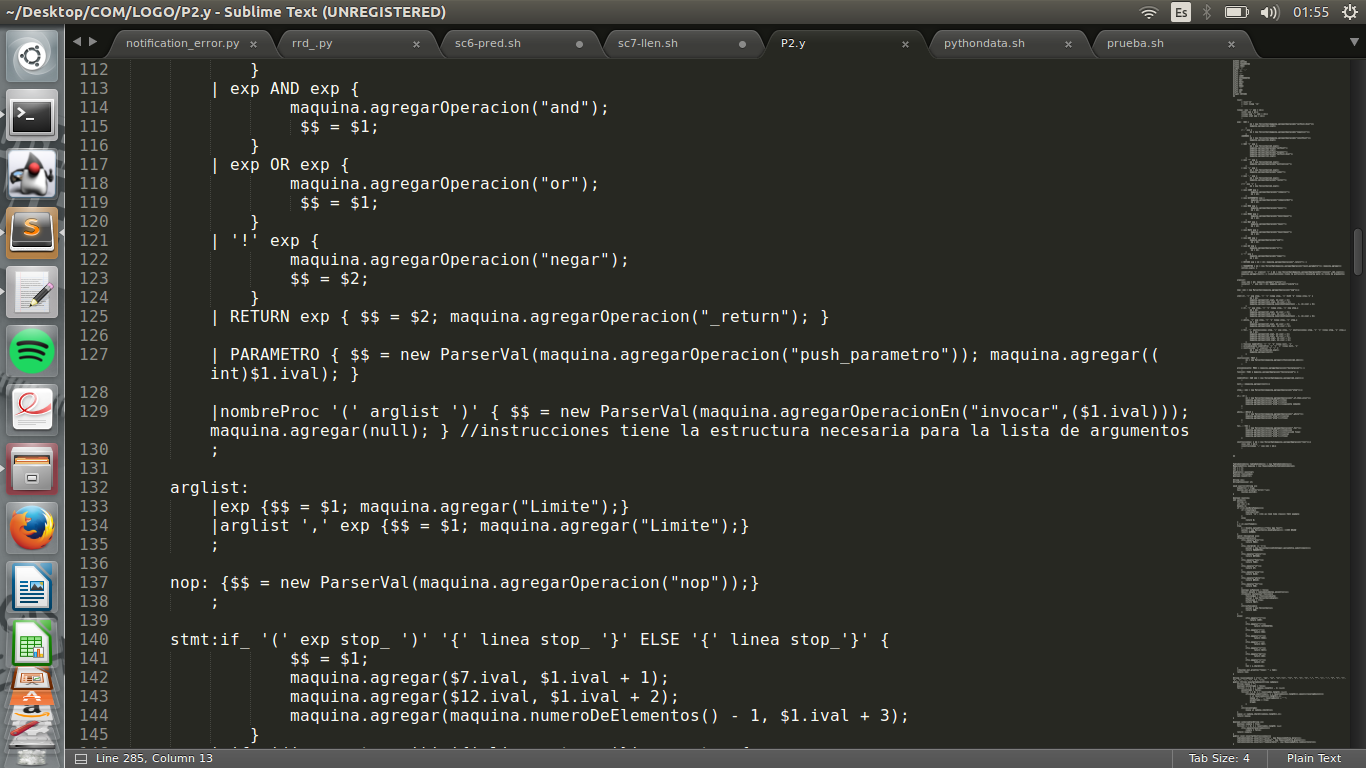
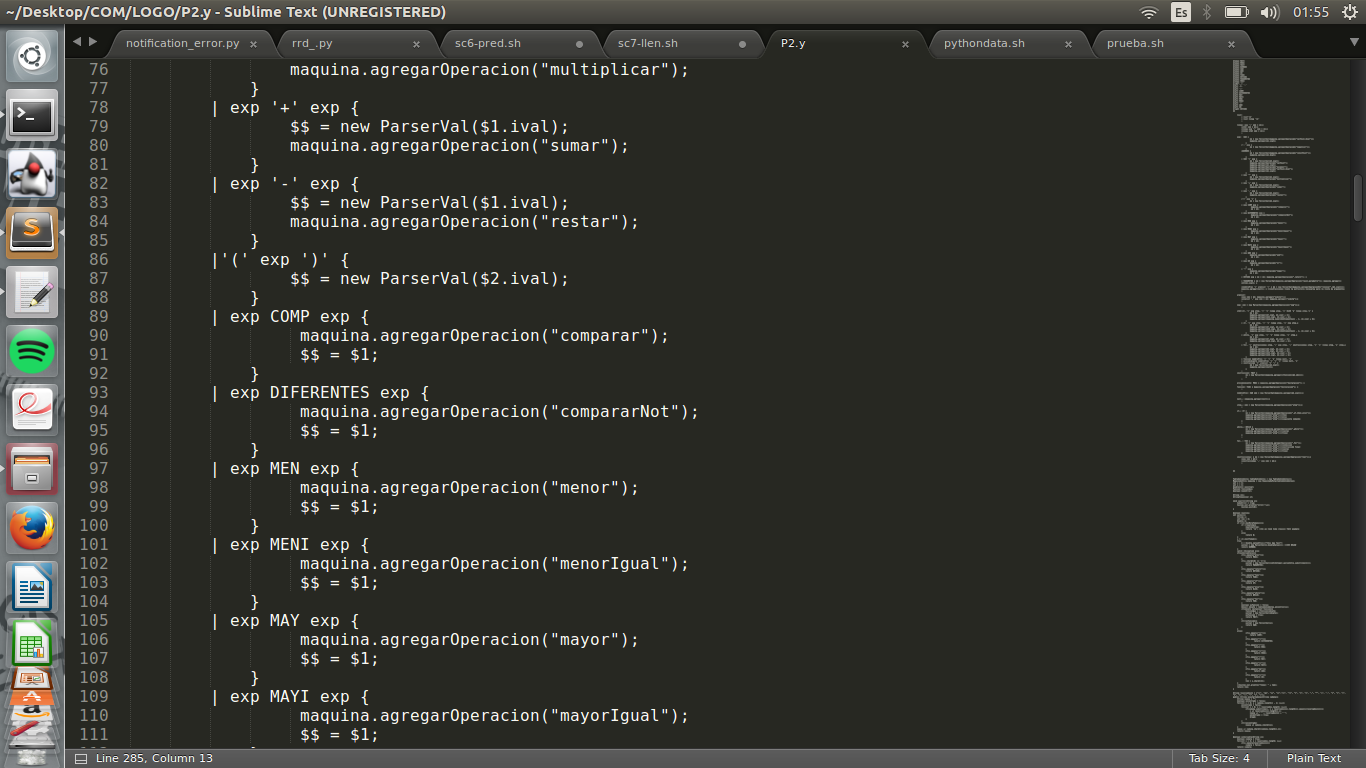
|  |  |
| --- | --- |
| **Función** | **Descripción** |
| *CambiarColor [ int, int , int ]* | Cambia el color de la linea a dibujar. |
| *Avanzar [ int ]* | Avanza el vector *int* unidades en dirección de la punta de la flecha |
| *Girar [ int ]* | Gira el vector *int* grados. |

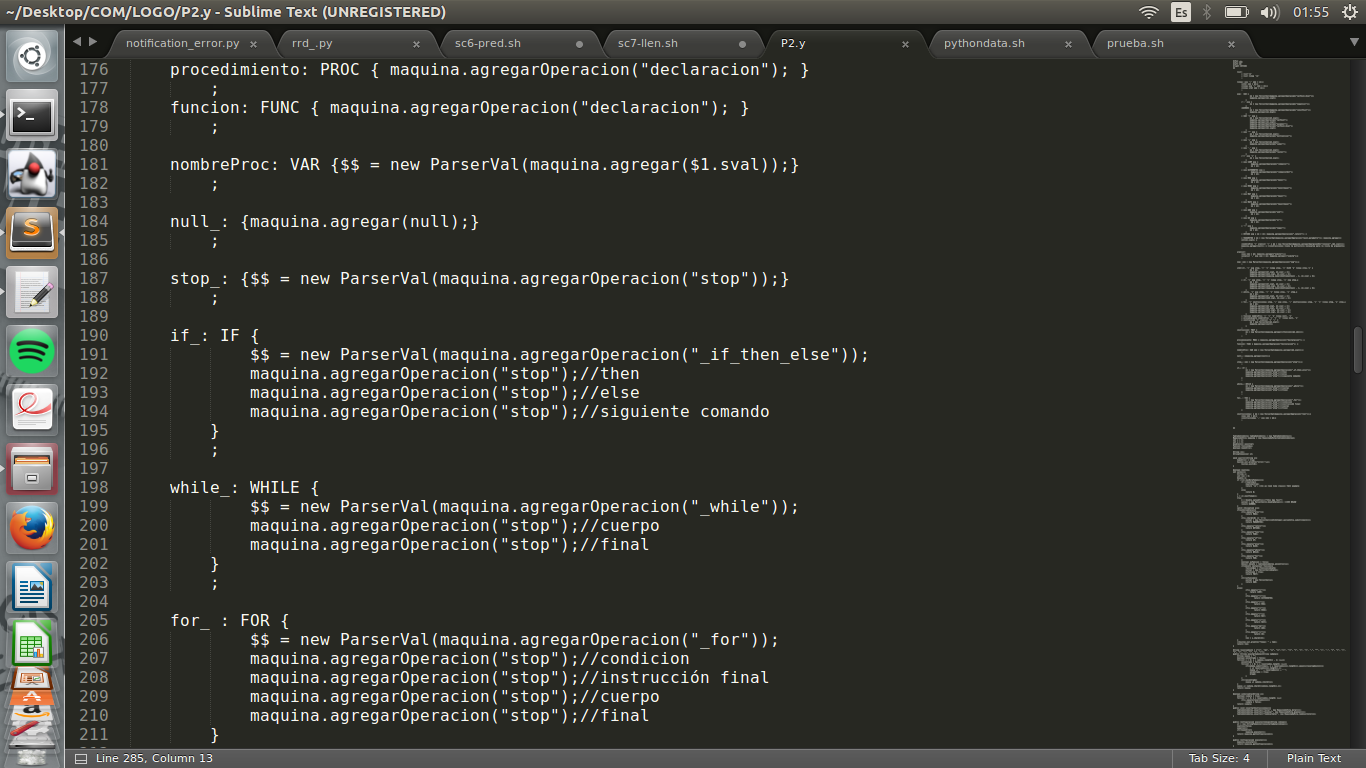
**CODIGO**

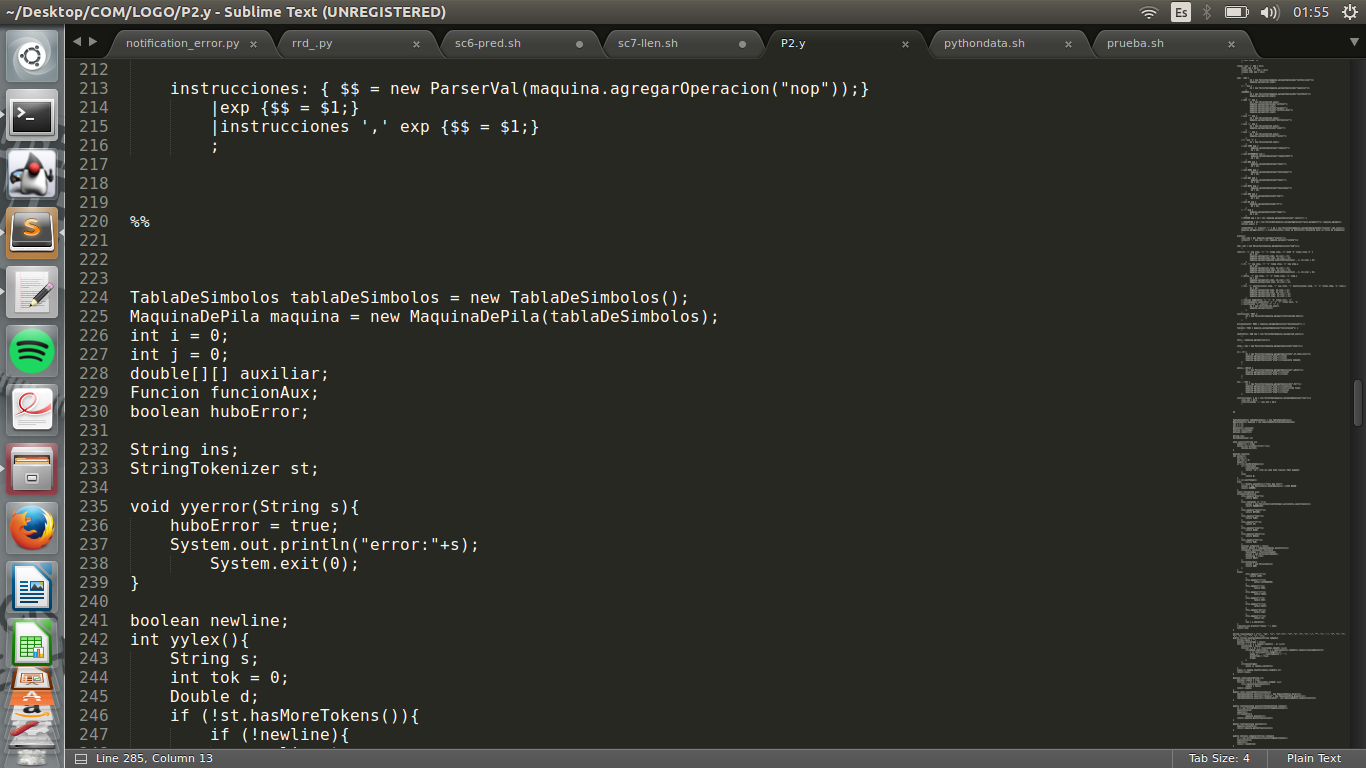
Para entender el uso del programa explicaremos las dos partes más importantes del código, la primera parte es el archivo .y el cuál es el que interpreta la sintaxis que el usuario inserta y define las acciones semánticas a ser ejecutadas, para ello tenemos los siguientes terminales definidos:



Y como producciones:





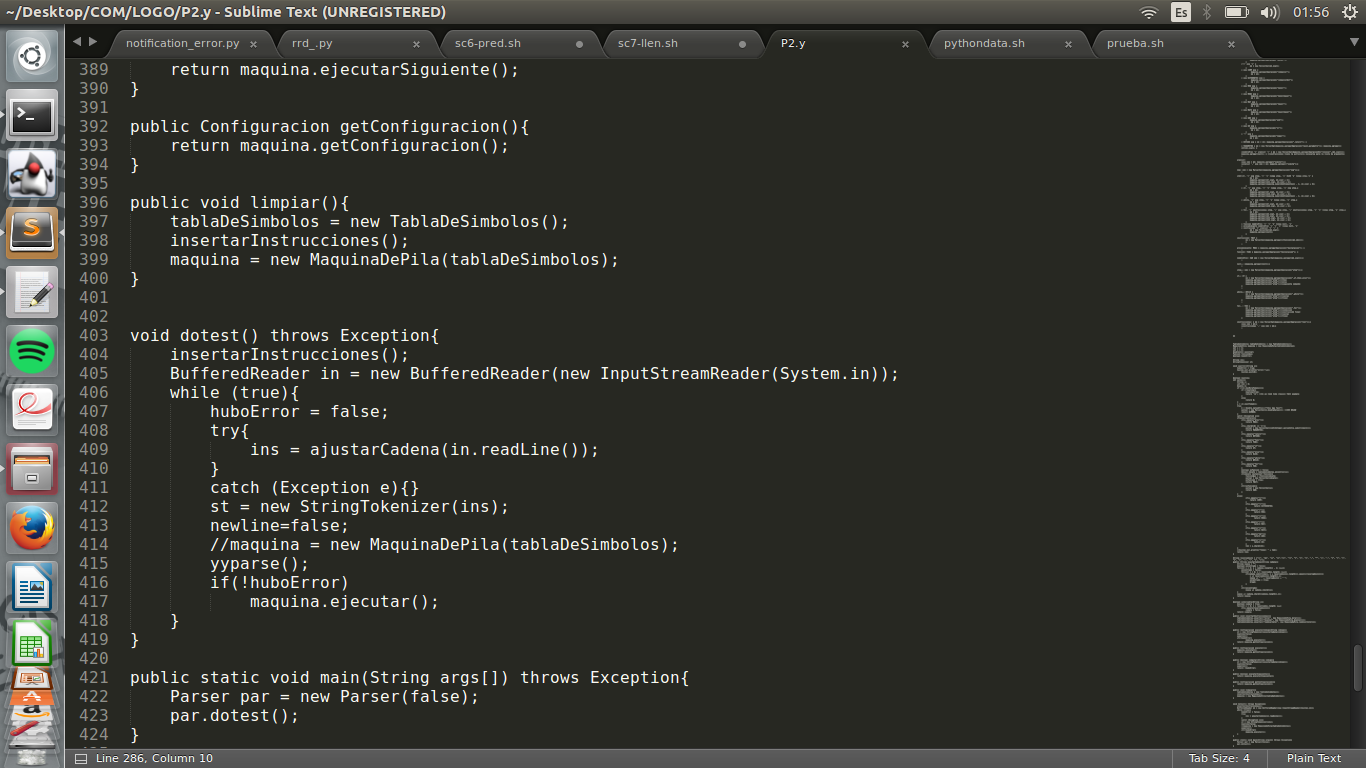
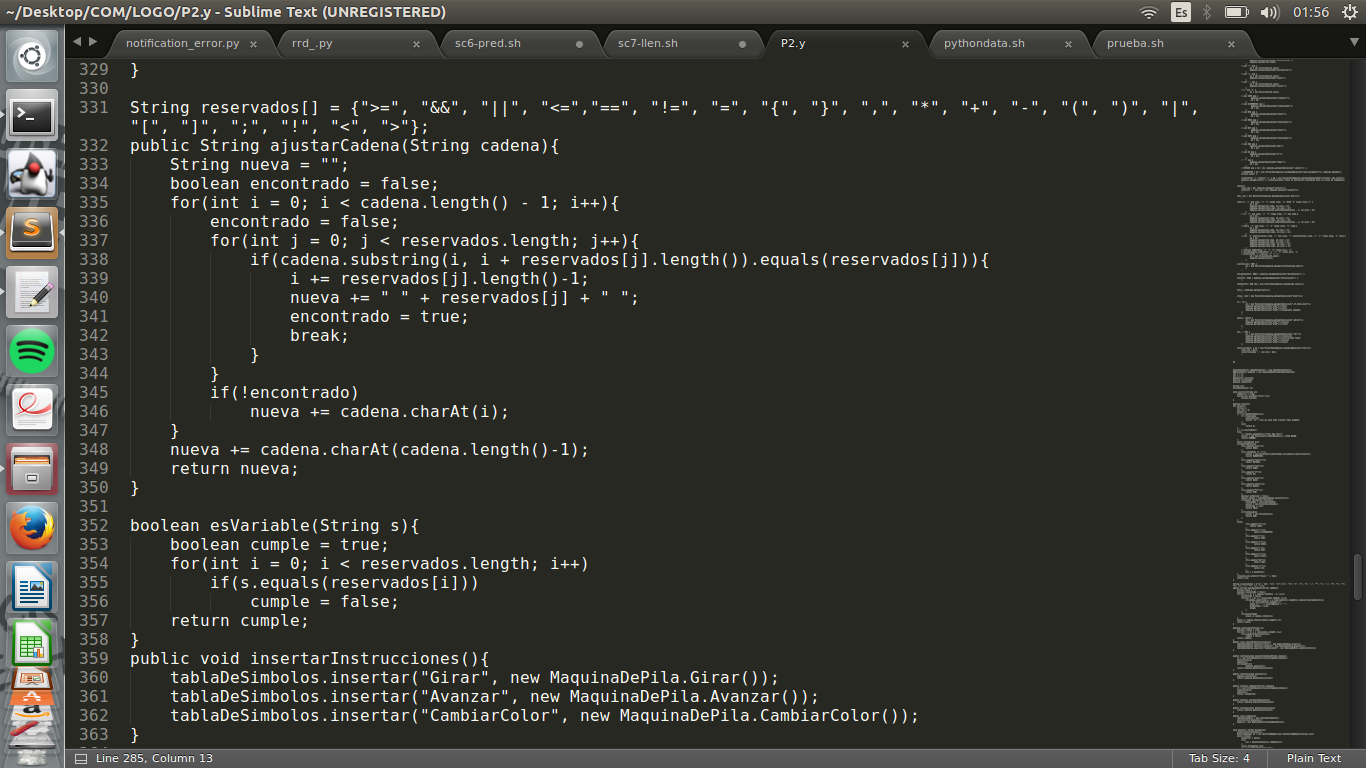
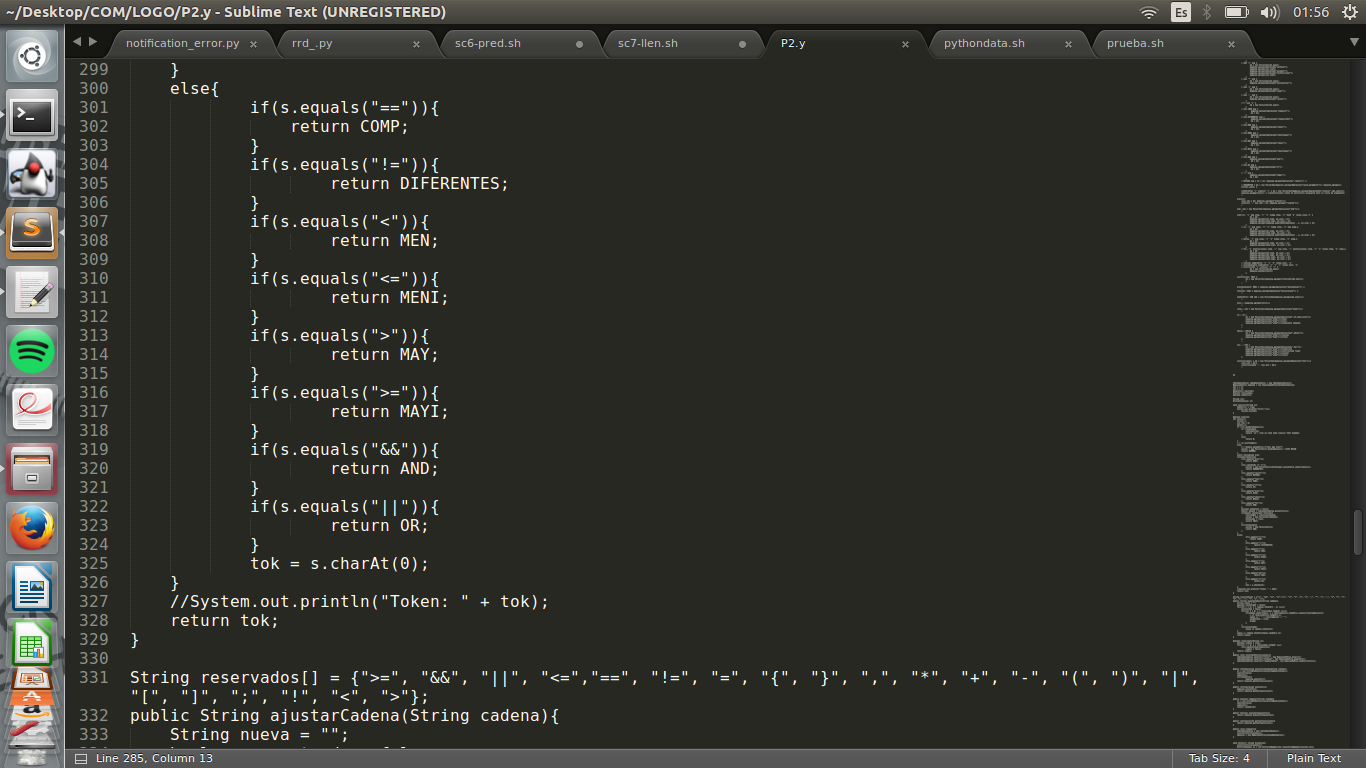
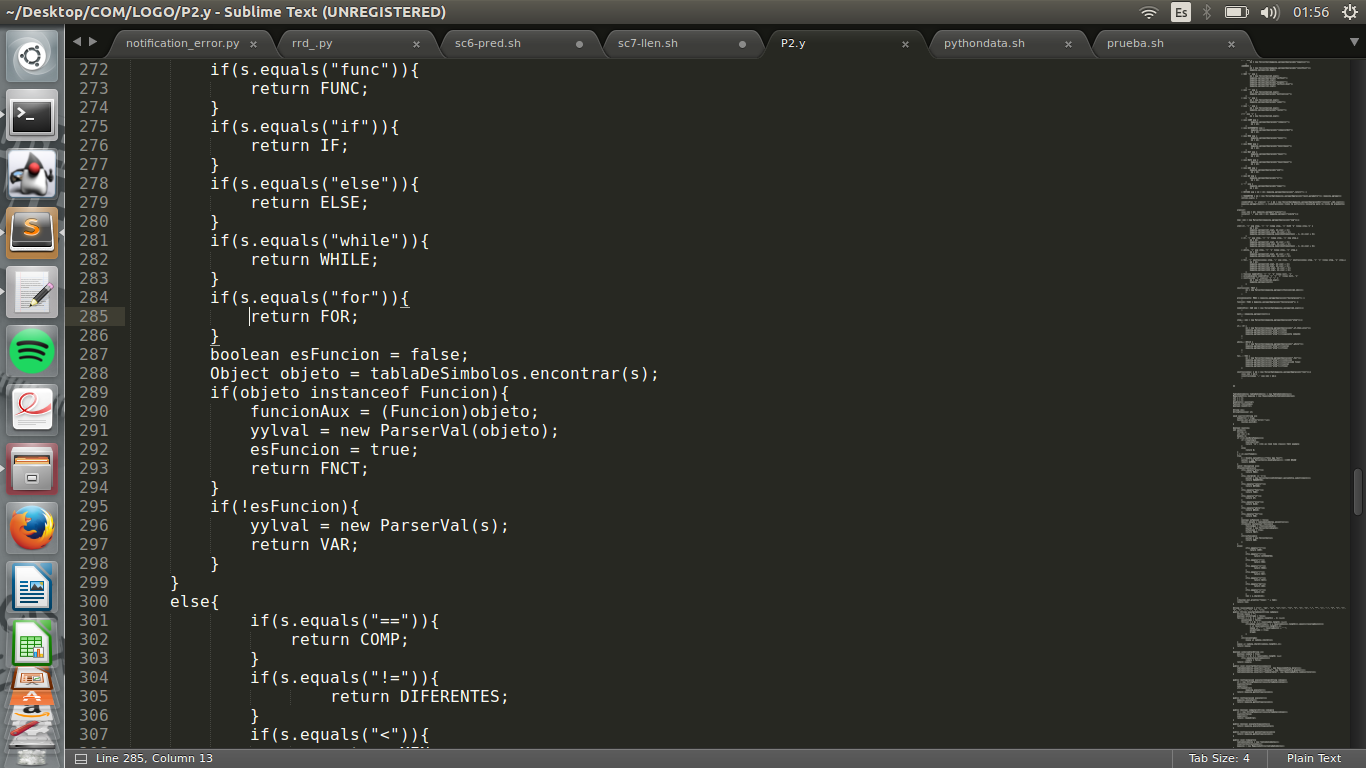


Hasta este punto hemos definido una lista de expresiones conformadas por, condiciones, ciclos y variables, escribimos las acciones semánticas que interactúan con la pila para el paso de argumentos y para colocar los STOPS necesarios.

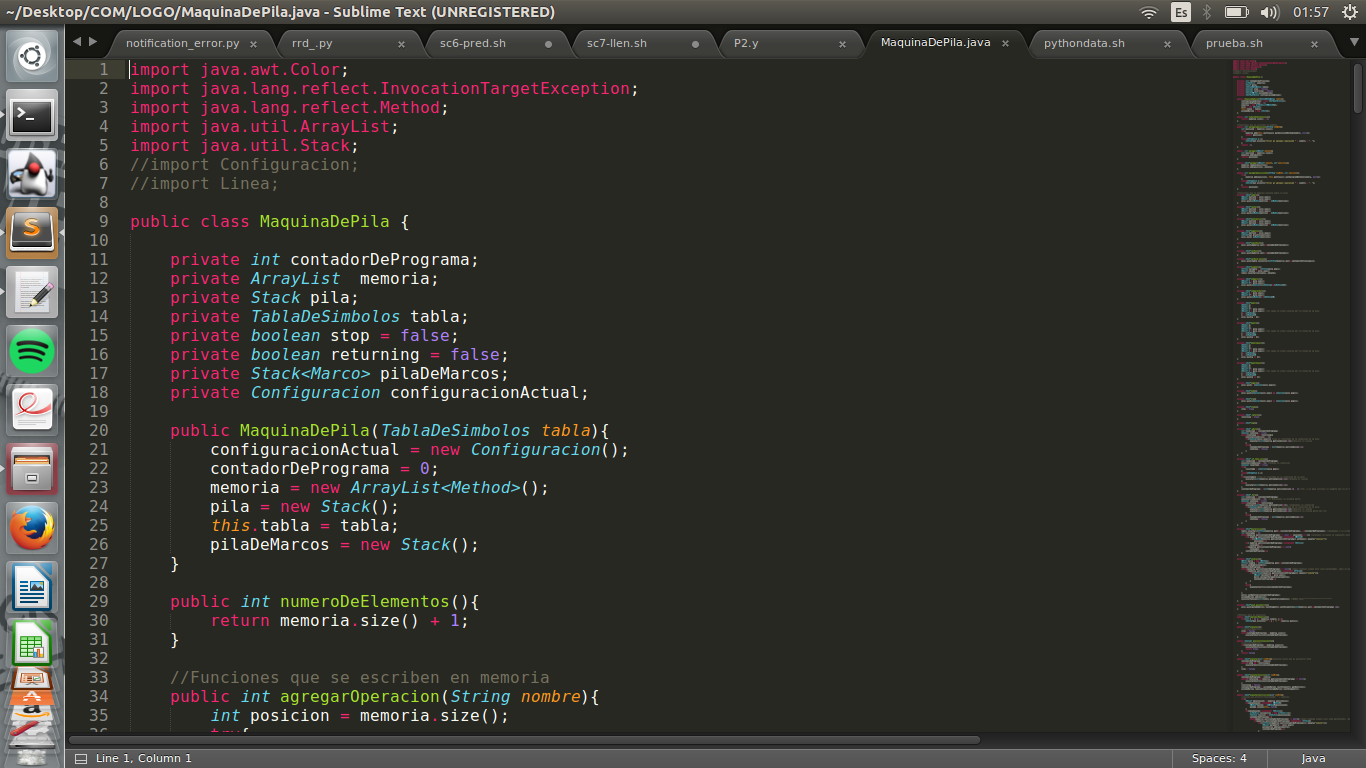
Ahora bien, es necesario escribir el código de yylex que se encarga de leer las líneas escritas para posteriormente dividirla en tokens que puedan ser procesados, esta es la función del analizador léxico:

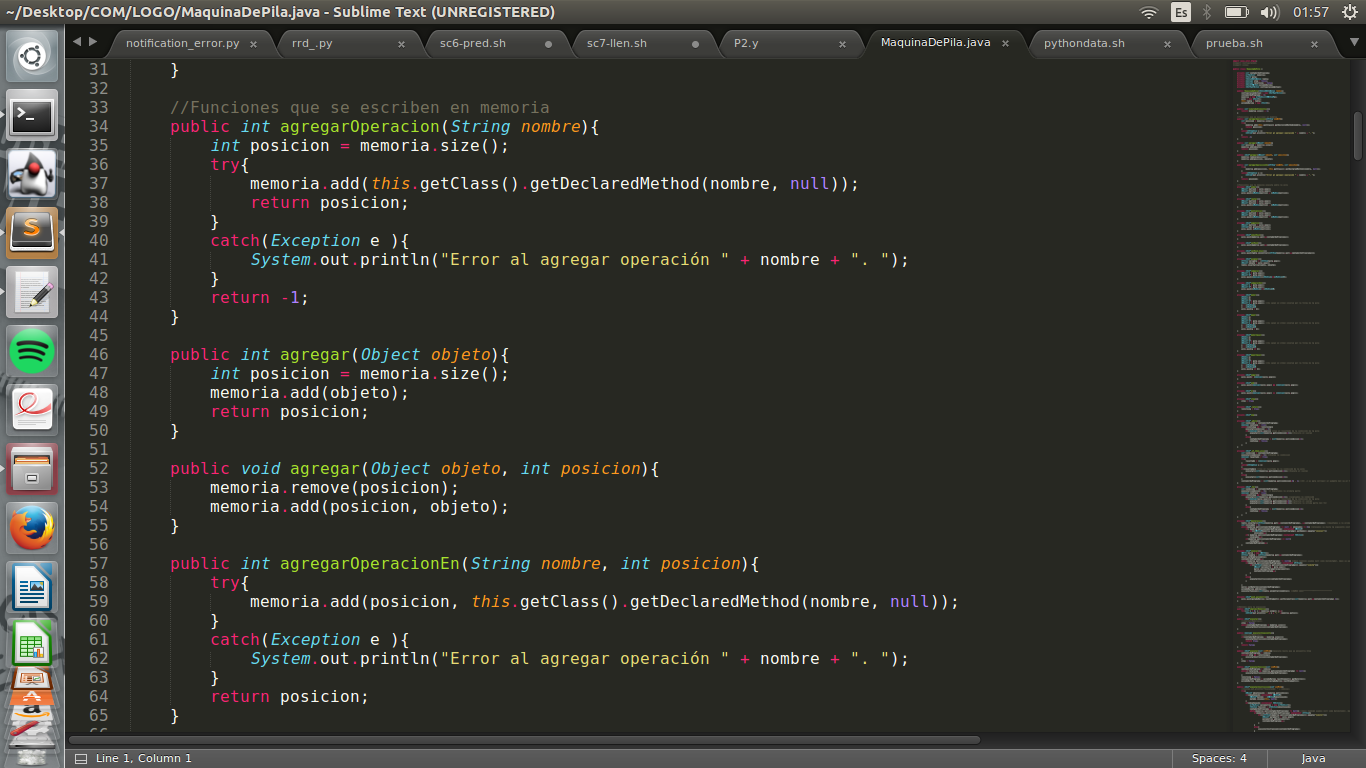
FUNCIONES ANALISIS LEXICO  
- Eliminación de espacios en blanco  
- Reconocimiento de identificadores y palabras clave  
- Reconocimiento de constantes   
- Mejorar la portabilidad del compilador

Posteriormente se hace un ANALISIS SINTACTICO  
Esta fase se puede practicar, siempre y cuando la anterior haya sido realizada correctamente, en este analizador se agrupan los componentes para construir frases, verifica que el lenguaje fuente cumpla con lasespecificaciones que necesita el compilador donde se va a ejecutar. Es la fase más importante en el proceso de compilación.

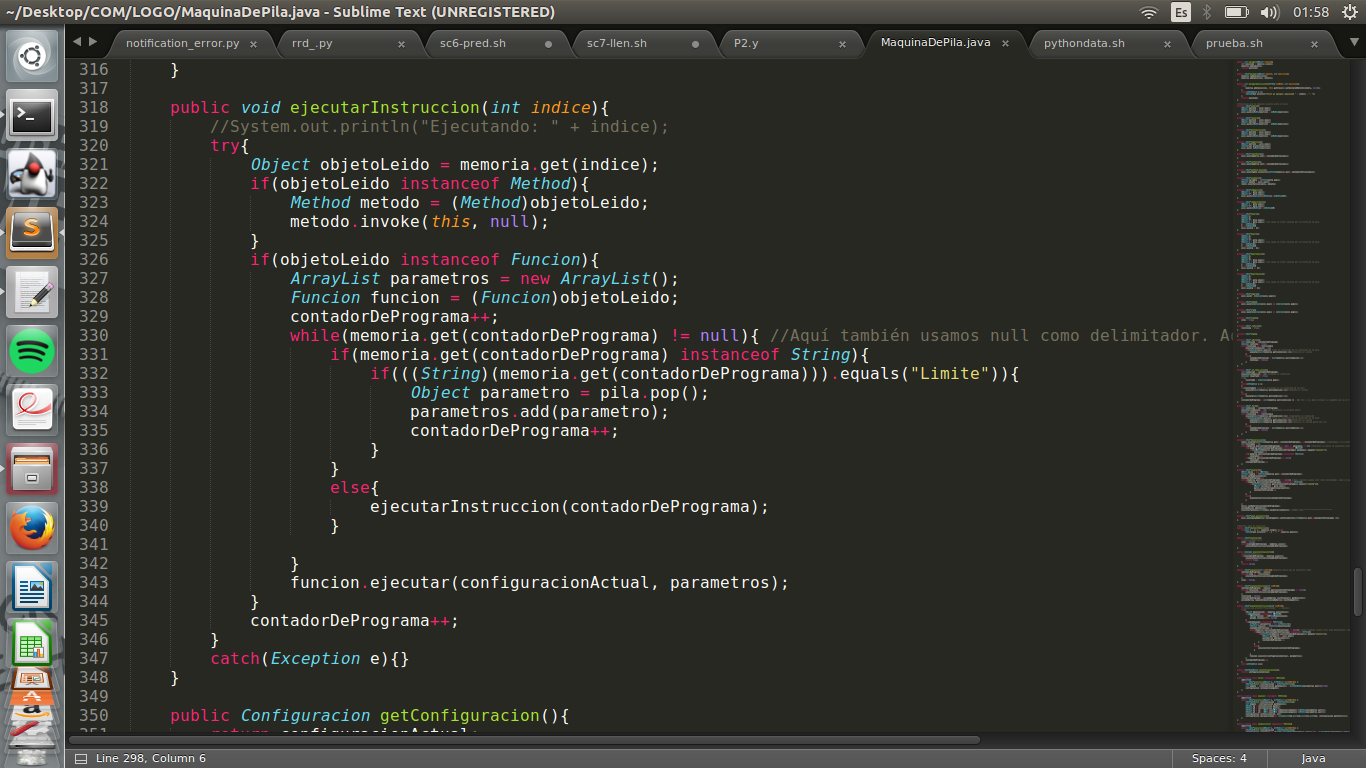
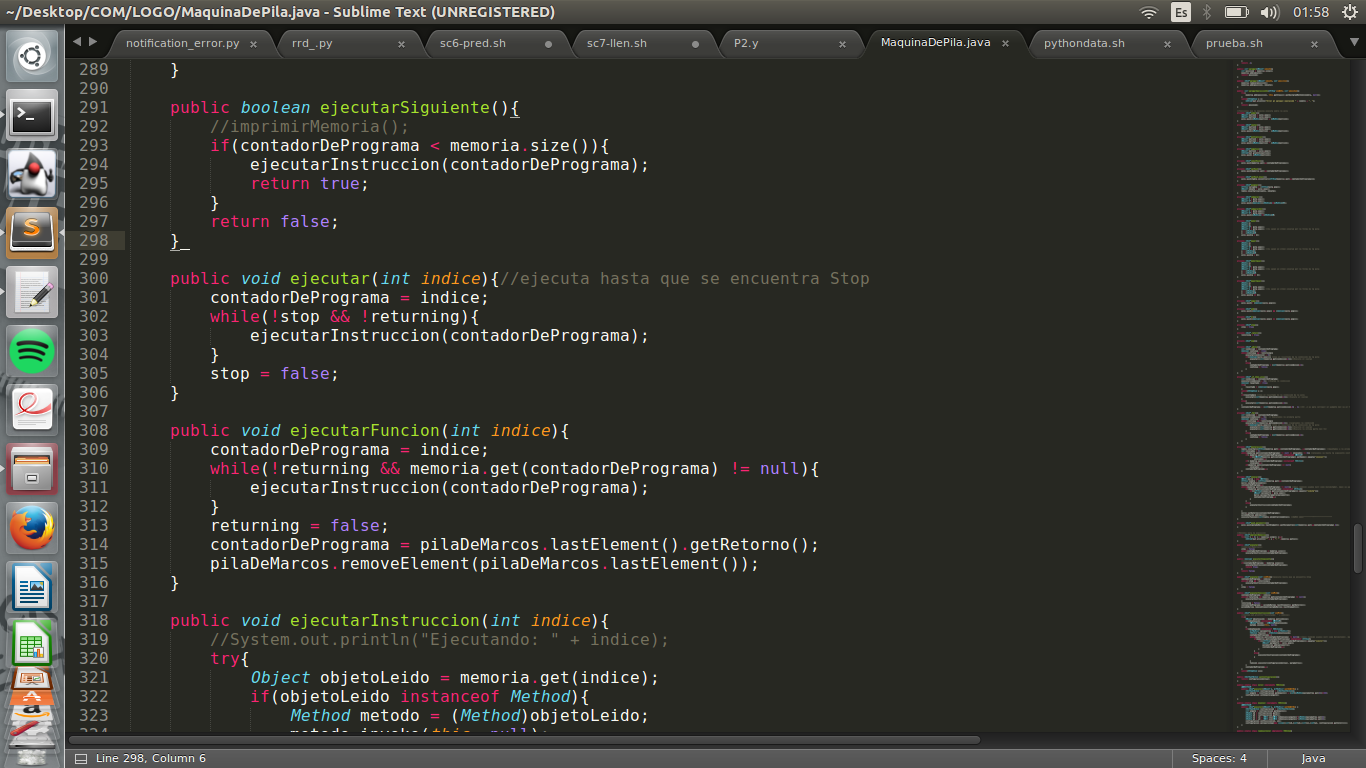
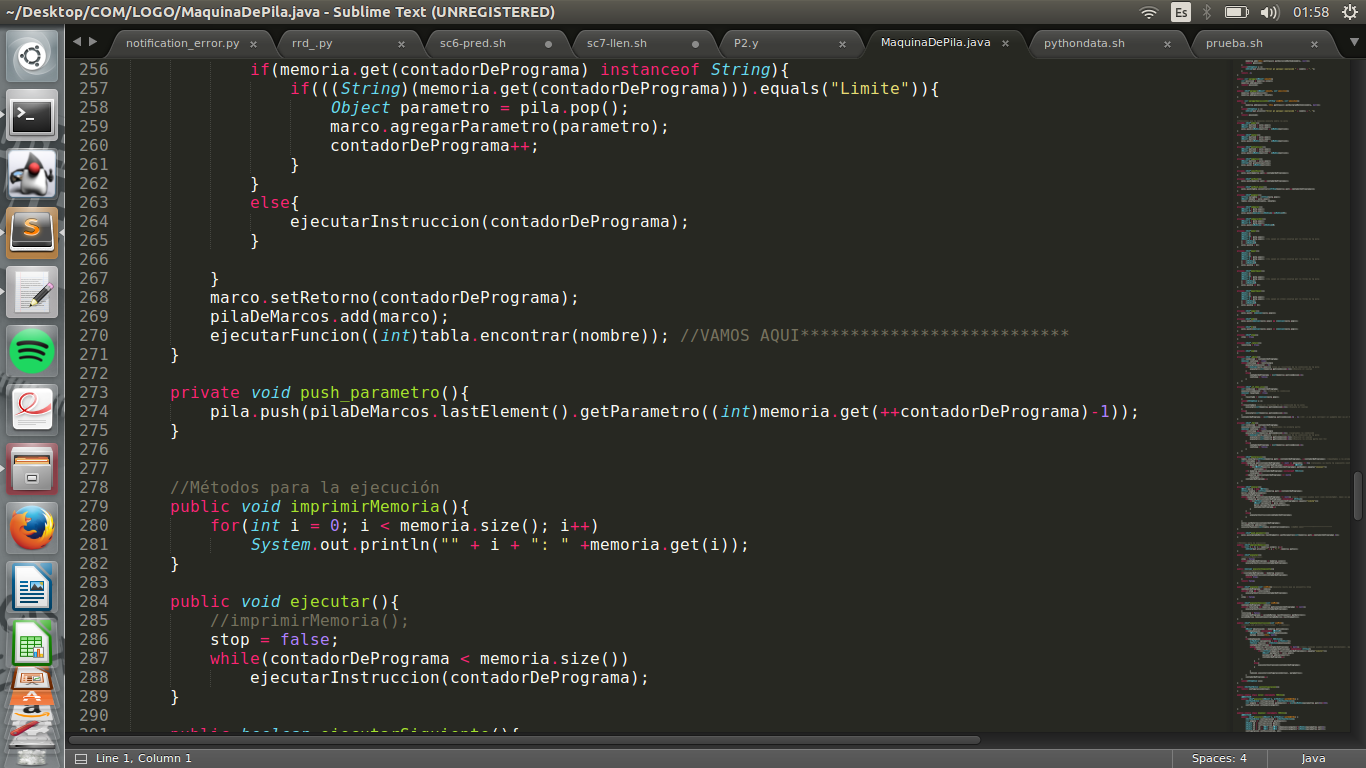
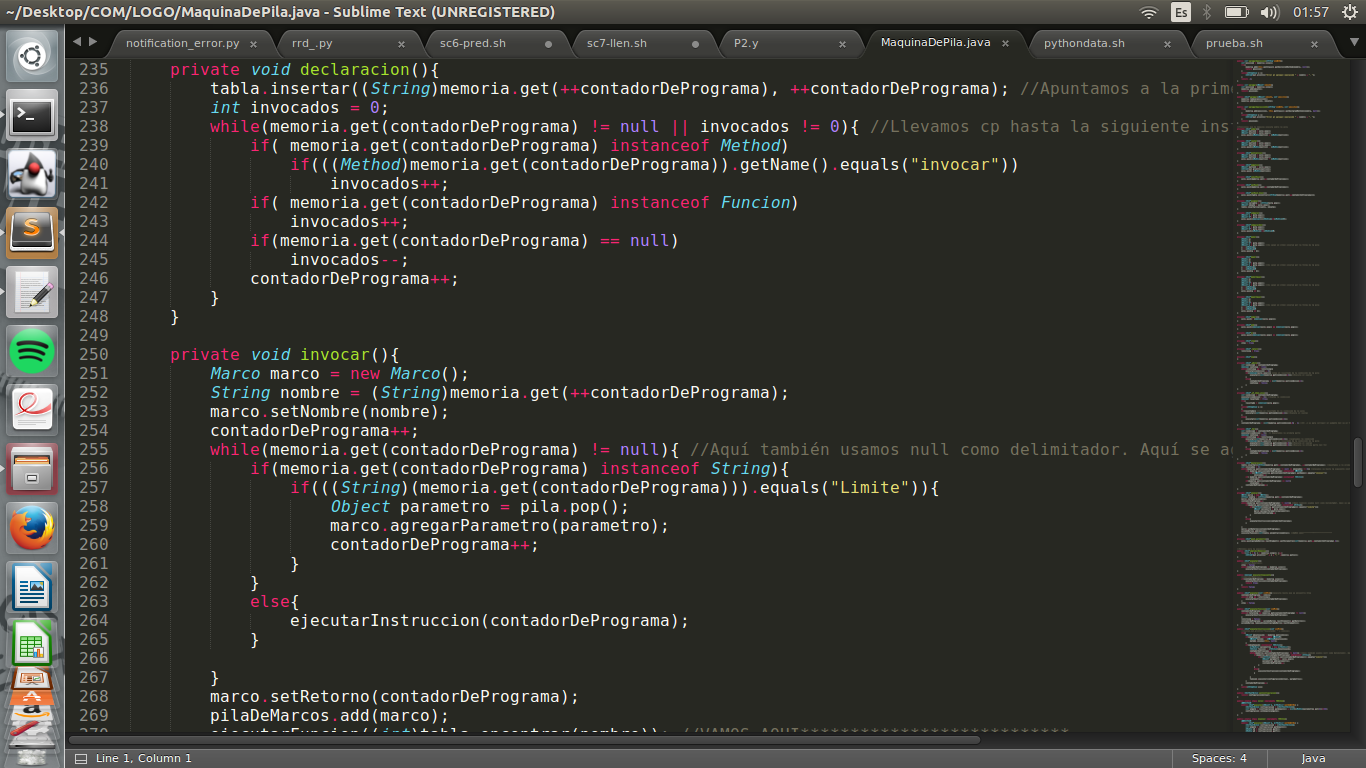
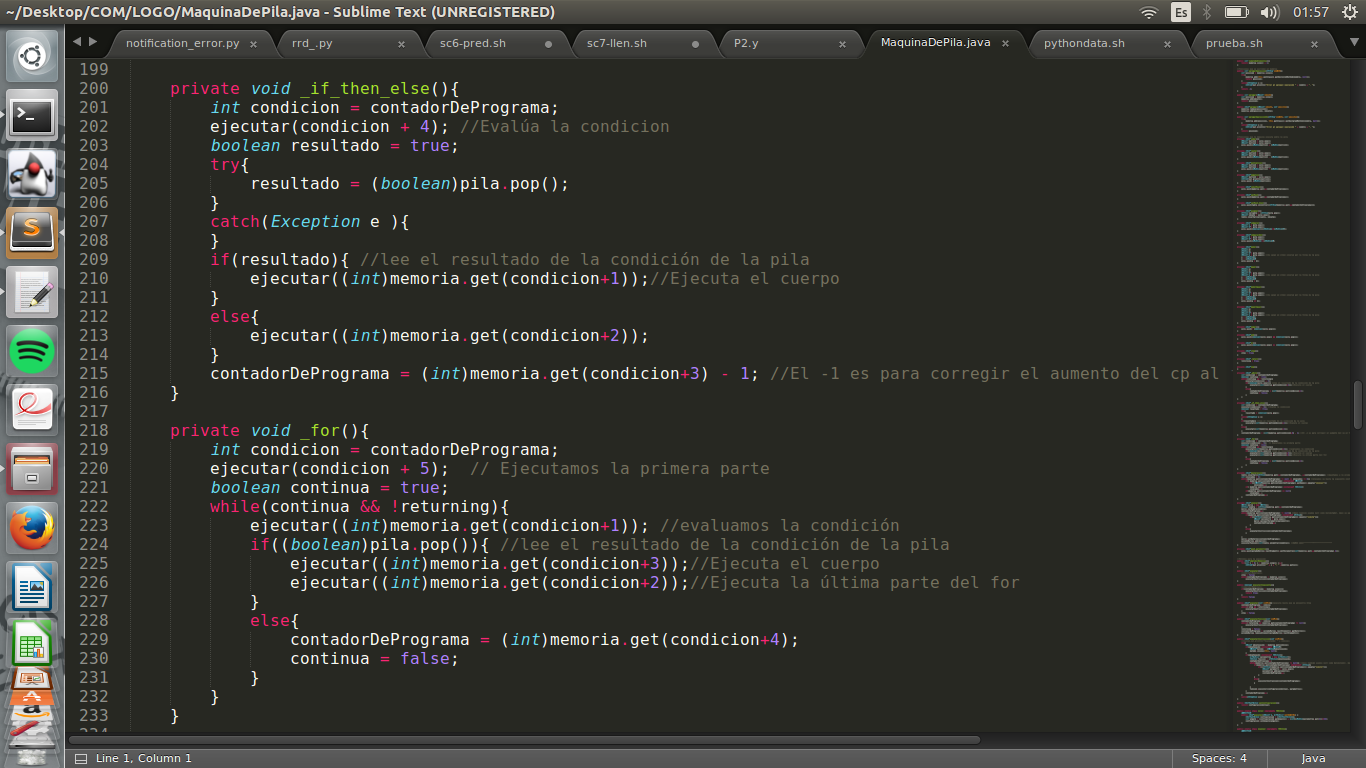
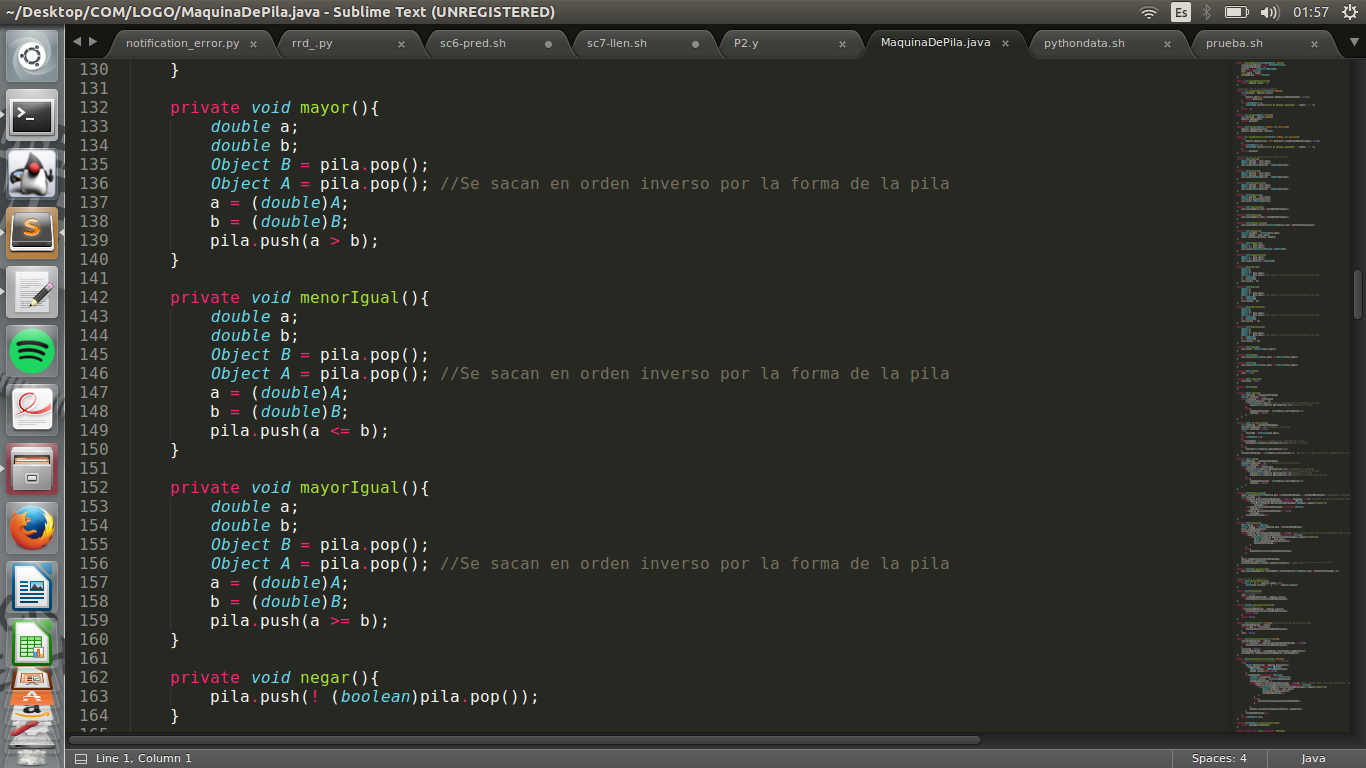
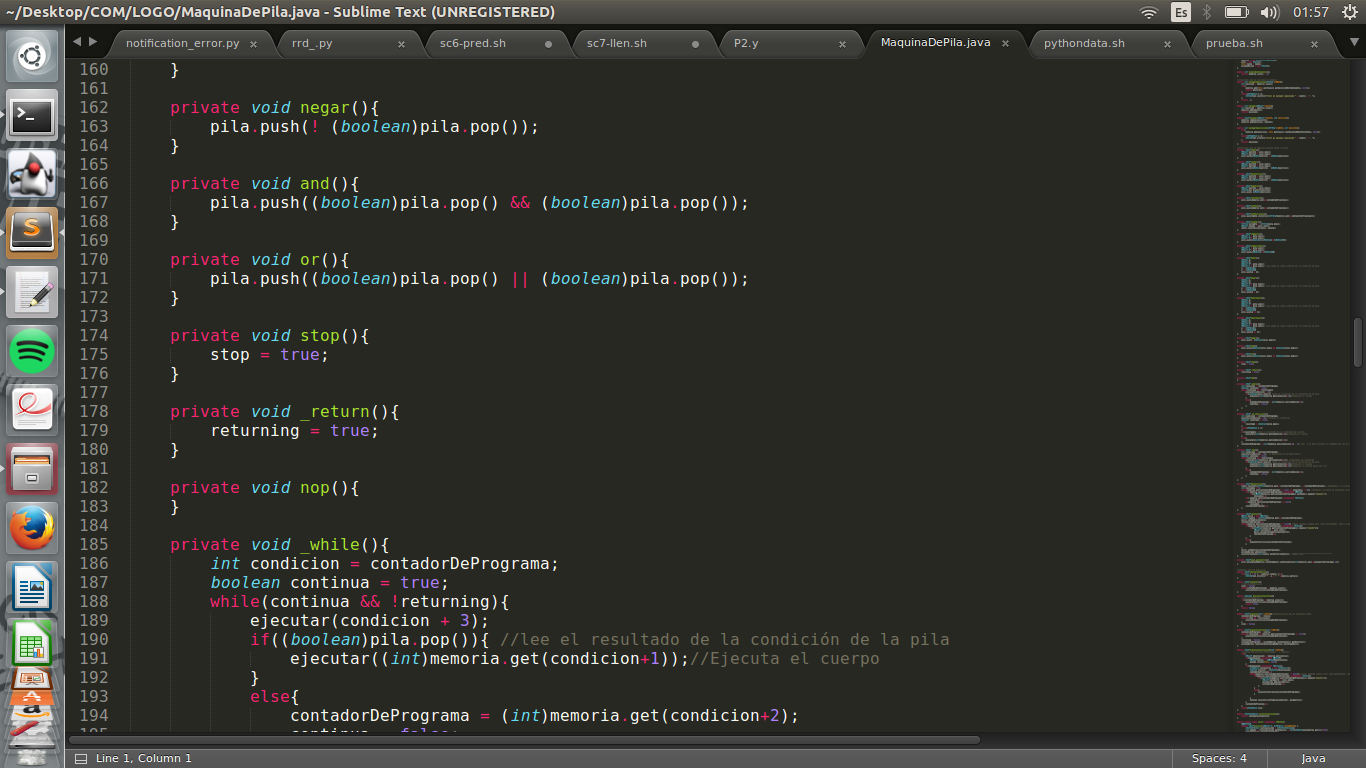
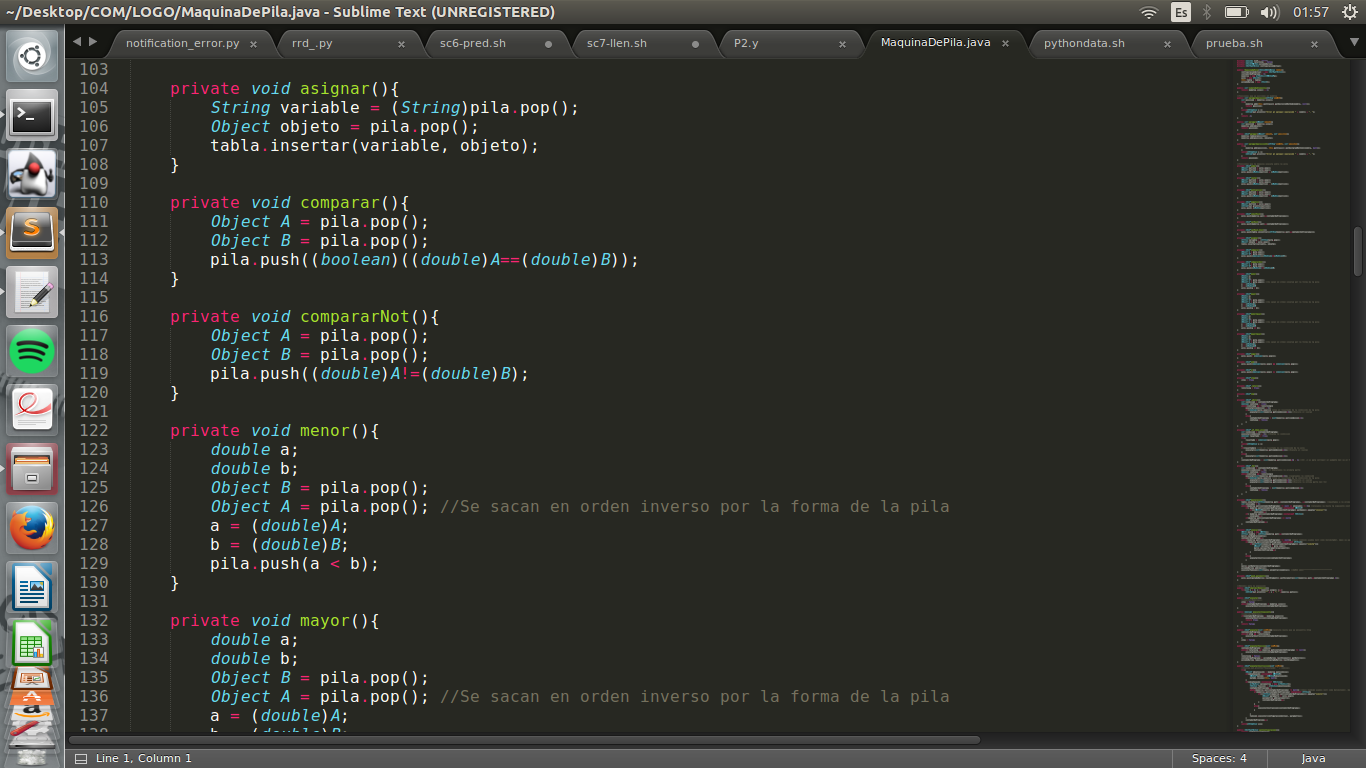
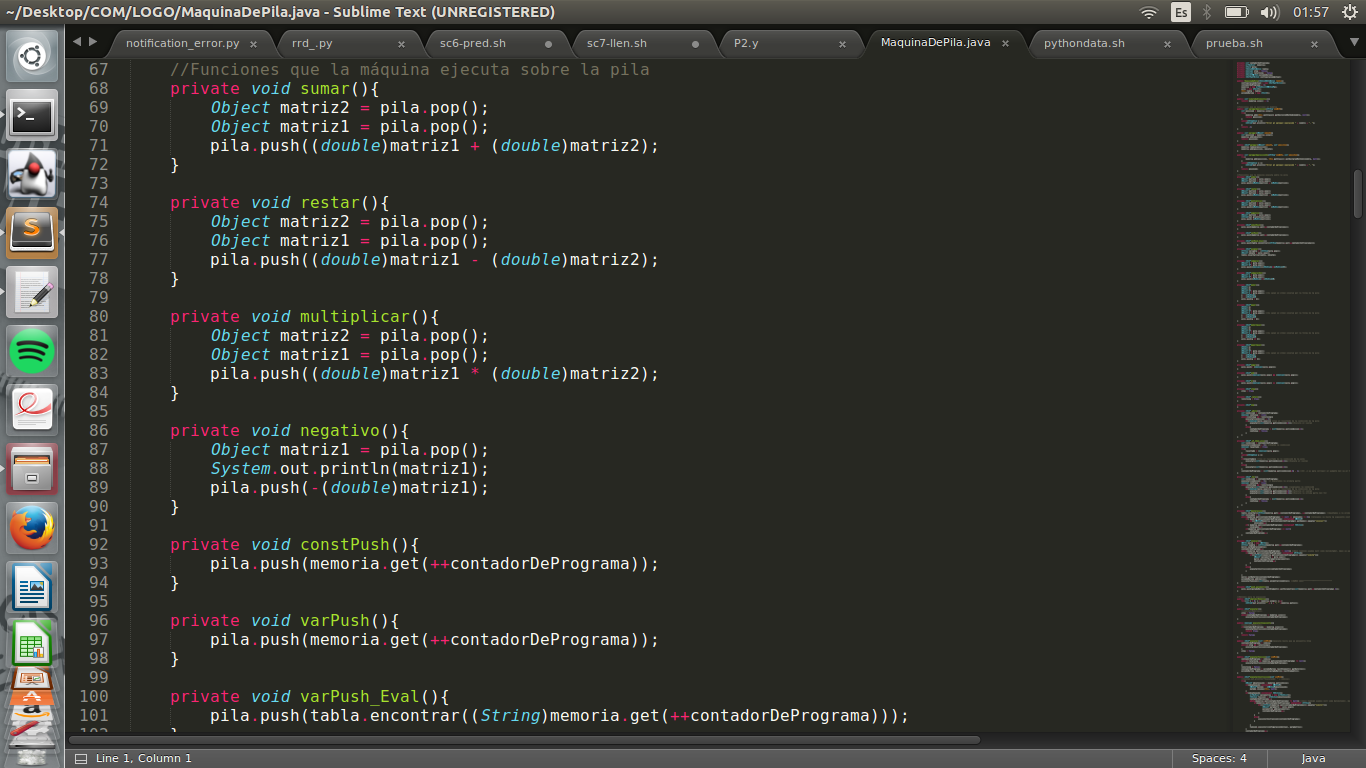


Para la siguiente etapa es momento de generar código con la utilización de la Máquina de Pila, la pila dedicada al Marco de las funciones , la pila auxiliar para almacenar los valores de las operaciones realizadas y la pila de YACC.





En la Maquina de pila se tienen las funciones principales que fueron llamadas desde el vector que simula la RAM, este código de la RAM se generó desde el archivo. y , y la Máquina es la encargada de ejecutarlo mediante el uso de reflexión .



La máquina de pila también usa una tabla de símbolos que se encargará de llevar la cuenta de que variables han sido declaradas previamente para no repetirlas. También se realizan las funciones de ANALISIS SEMANTICO:  
- Dentro de sus funciones está la de comprobar que el significado de lo que se va leyendo sea válido.  
- Debe verificar que los caracteres sean compatibles con el operador que se va a utilizar.

Finalmente con todas estas herramientas se obtiene la etapa de GENERACION CODIGO INTERMEDIO, los compiladores deben analizar y generar secuencias de instrucciones iguales a un programa fuente con el fin de aprovechar mucho más la memoria del equipo, en esta etapa se genera un código con el fin de que este sea optimizando en la siguiente etapa.

# **REFERENCIAS**

https://logo.codeplex.com

*Logo Lenguaje de Programacion*. Obtenido de Wikipedia: https://es.wikipedia.org/wiki/Logo\_(lenguaje\_de\_programaci%C3%B3n)