Instituto Tecnológico de Chilpancingo

Ingeniería en sistemas computacionales

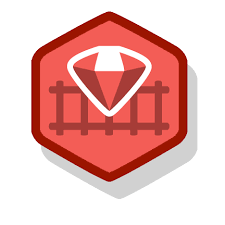
**Lenguajes y Autómatas II**

Catedrático:

**Alfredo de Jesús Canto Cetina**

Trabajo:

“Analizador Recursivo en Ruby”



Integrantes del Equipo:

* **Cynthia Daniela García González**
* **Abigail Mosso Martínez**
* **Guillermo Peña Figueroa**

Índice

*Contenido:*

Introducción………………………………………………………….………………pag 3

Autómatas…………………………………………………………….………………pag 4

Diagramas de flujo…………………………………………………….pag 5 – pag 6

Codificación en Ruby………………………………………………..…...pag 7 - 14

Pruebas…………………………………………………………………pag 15 – pag 22

Introducción:

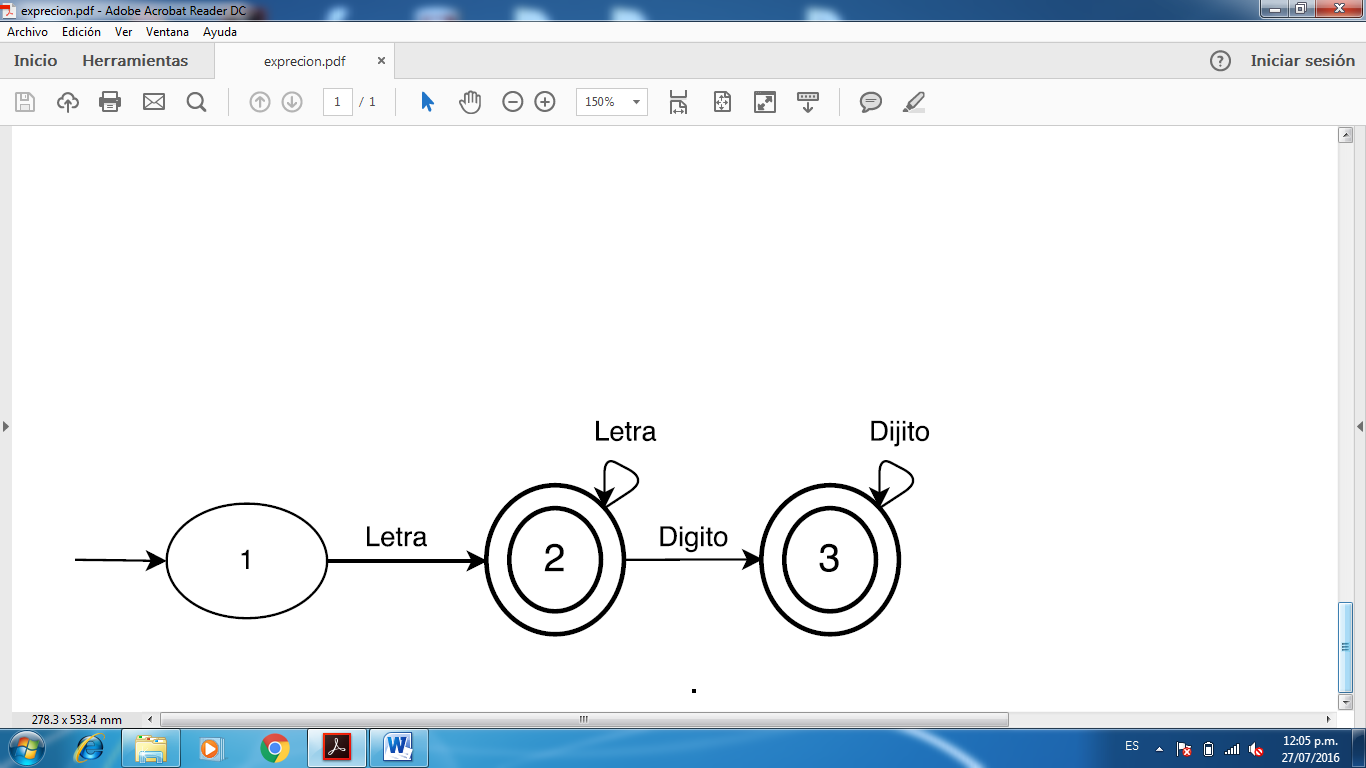
A lo largo del curso de verano, aprendimos a manejar Ruby. Ruby es un lenguaje de programación  [orientado a objetos](https://es.wikipedia.org/wiki/Orientado_a_objetos): todos los tipos de datos son un objeto, incluidas las clases y tipos que otros lenguajes definen como primitivas, (como [enteros](https://es.wikipedia.org/wiki/Entero), booleanos, y "nil"). Toda [función](https://es.wikipedia.org/wiki/Funci%C3%B3n_(programaci%C3%B3n)) es un [método](https://es.wikipedia.org/wiki/M%C3%A9todo_(programaci%C3%B3n_orientada_a_objetos)). Las variables siempre son referencias a objetos, no los objetos mismos. Ruby soporta [herencia](https://es.wikipedia.org/wiki/Herencia_(programaci%C3%B3n_orientada_a_objetos)) con [enlace dinámico](https://es.wikipedia.org/wiki/Enlace_din%C3%A1mico) y [métodos](https://es.wikipedia.org/wiki/Singleton)  (pertenecientes y definidos por una sola [instancia](https://es.wikipedia.org/wiki/Instancia_(programaci%C3%B3n)) más que definidos por la clase). A pesar de que Ruby no soporta [herencia múltiple](https://es.wikipedia.org/wiki/Herencia_m%C3%BAltiple), la clases pueden importar [módulos](https://es.wikipedia.org/wiki/M%C3%B3dulo_(programaci%C3%B3n)).

En este último programa programado en ruby, realizamos un analizador léxico recursivo, con la ayuda de los autómatas ya conocidos como: identificador y número real con exponente, así mismo de los diagramas de flujo de: Factor, termino, expresión y expresión simple. En el cual corroboramos su sintaxis correcta.

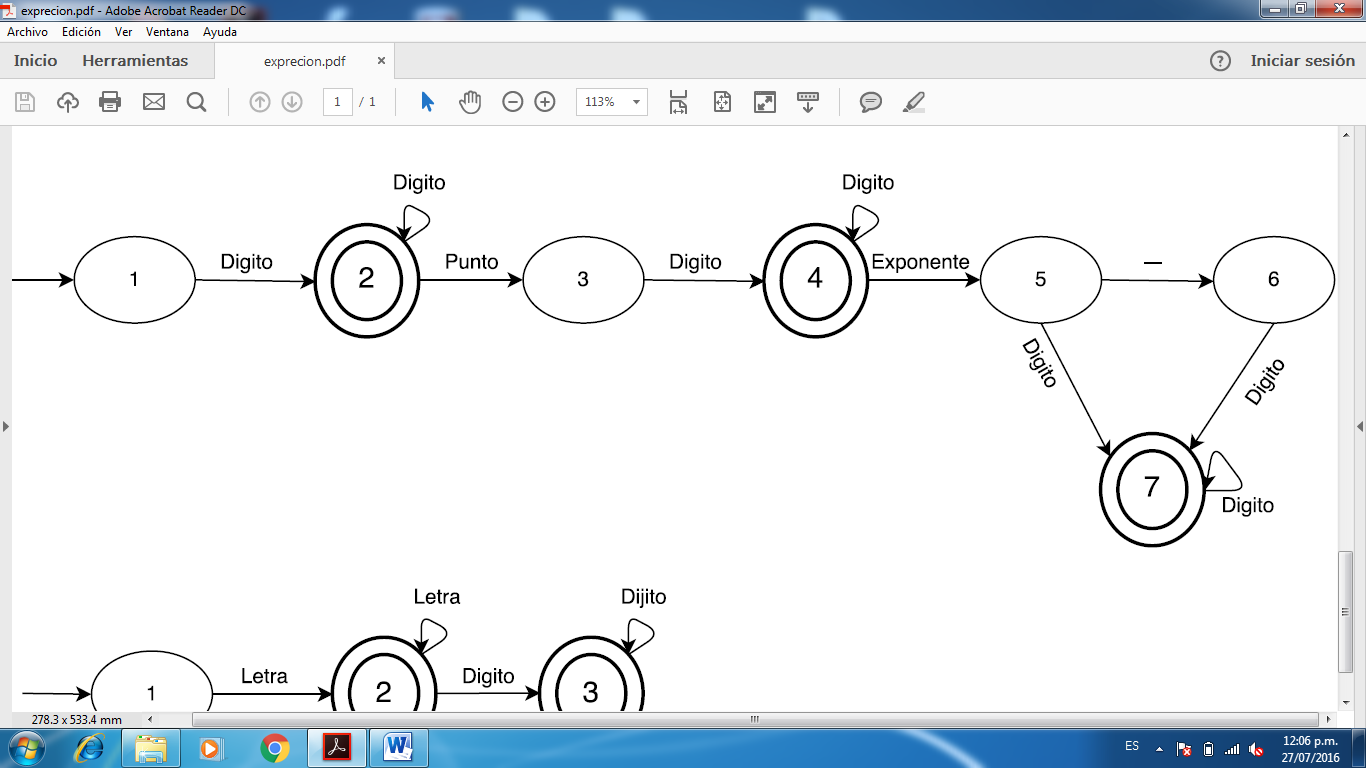


Autómatas

Autómata Identificador

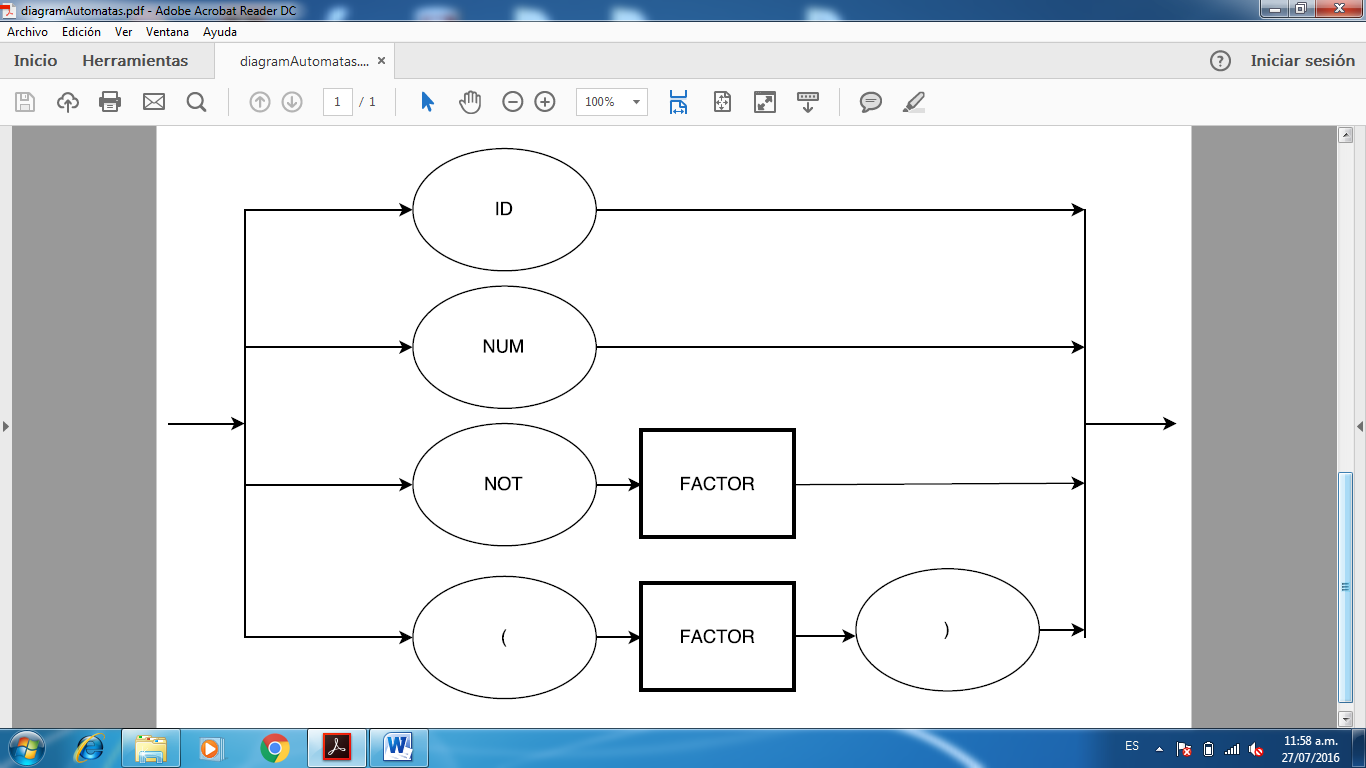


Autómata numero general (Numero, real y numero real con exponente)

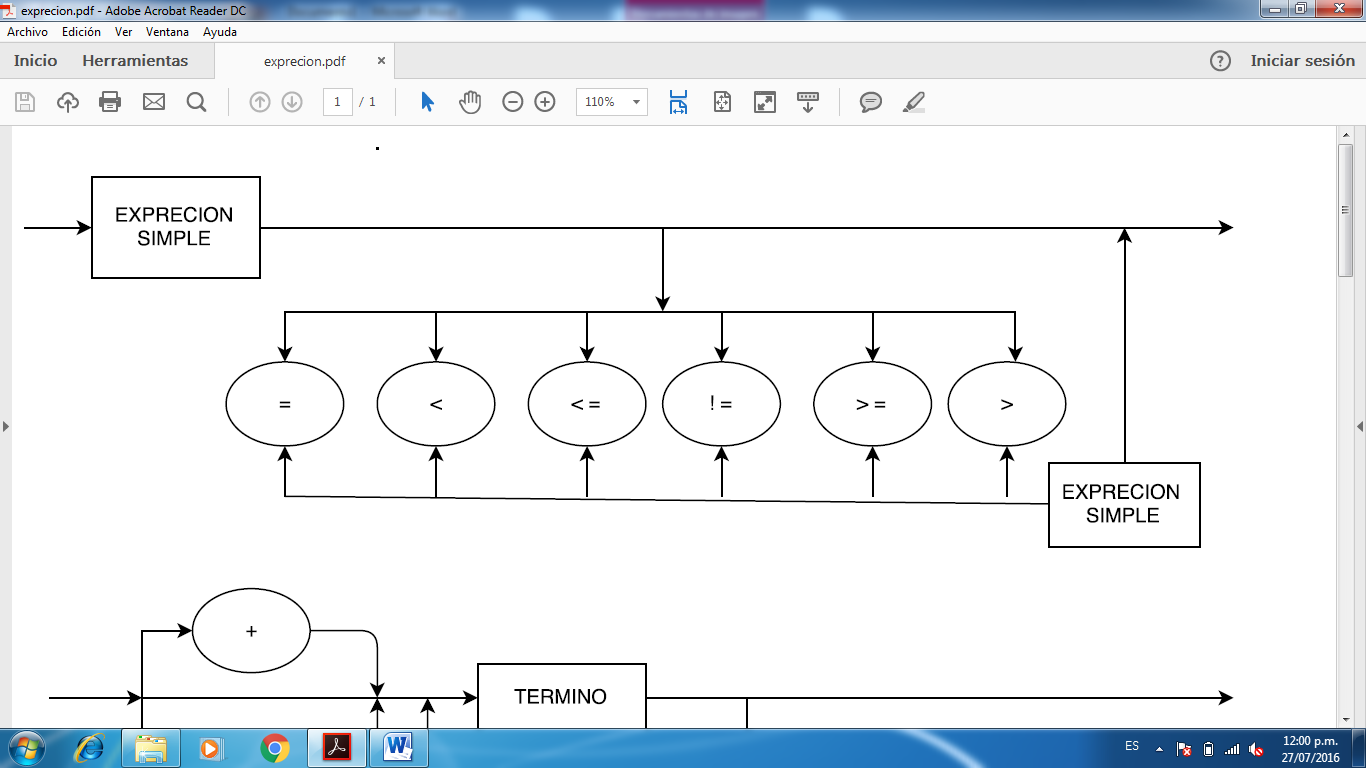


Diagramas de flujo

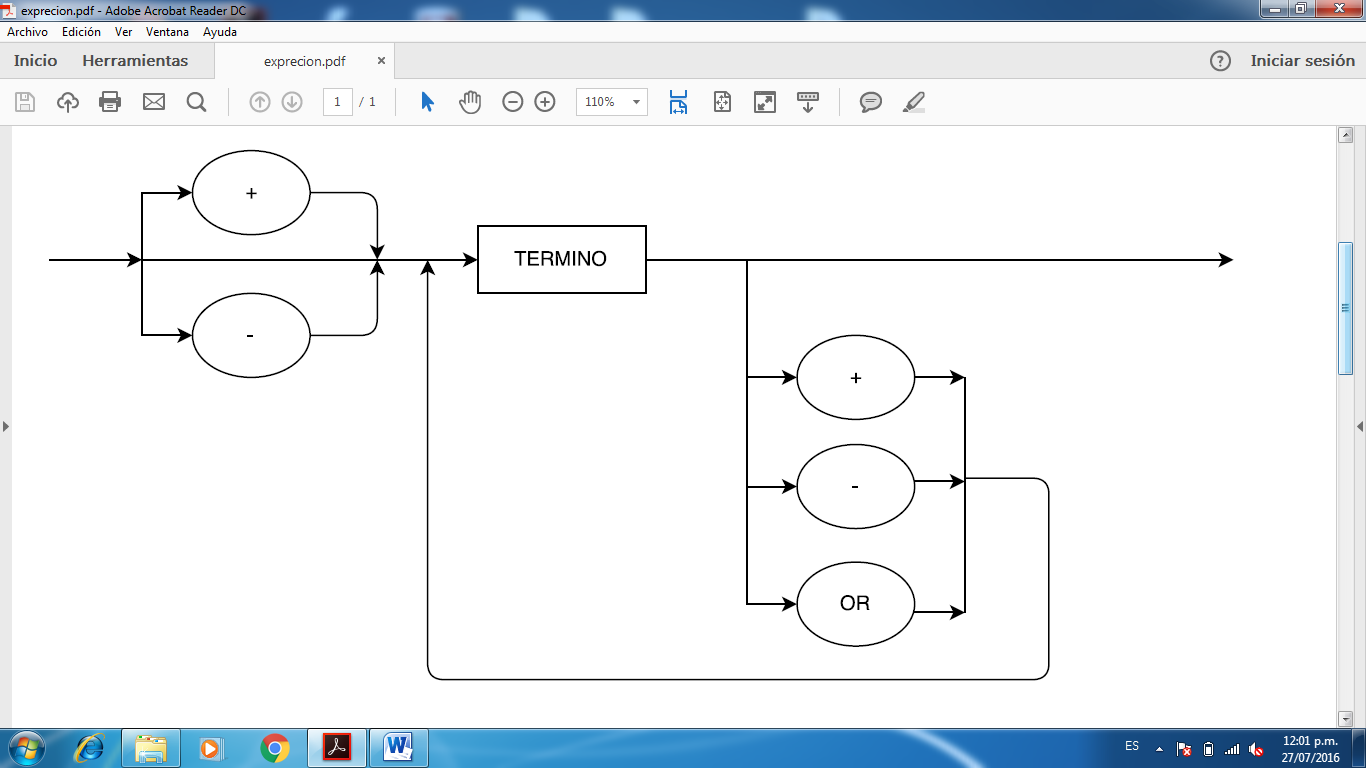
Factor



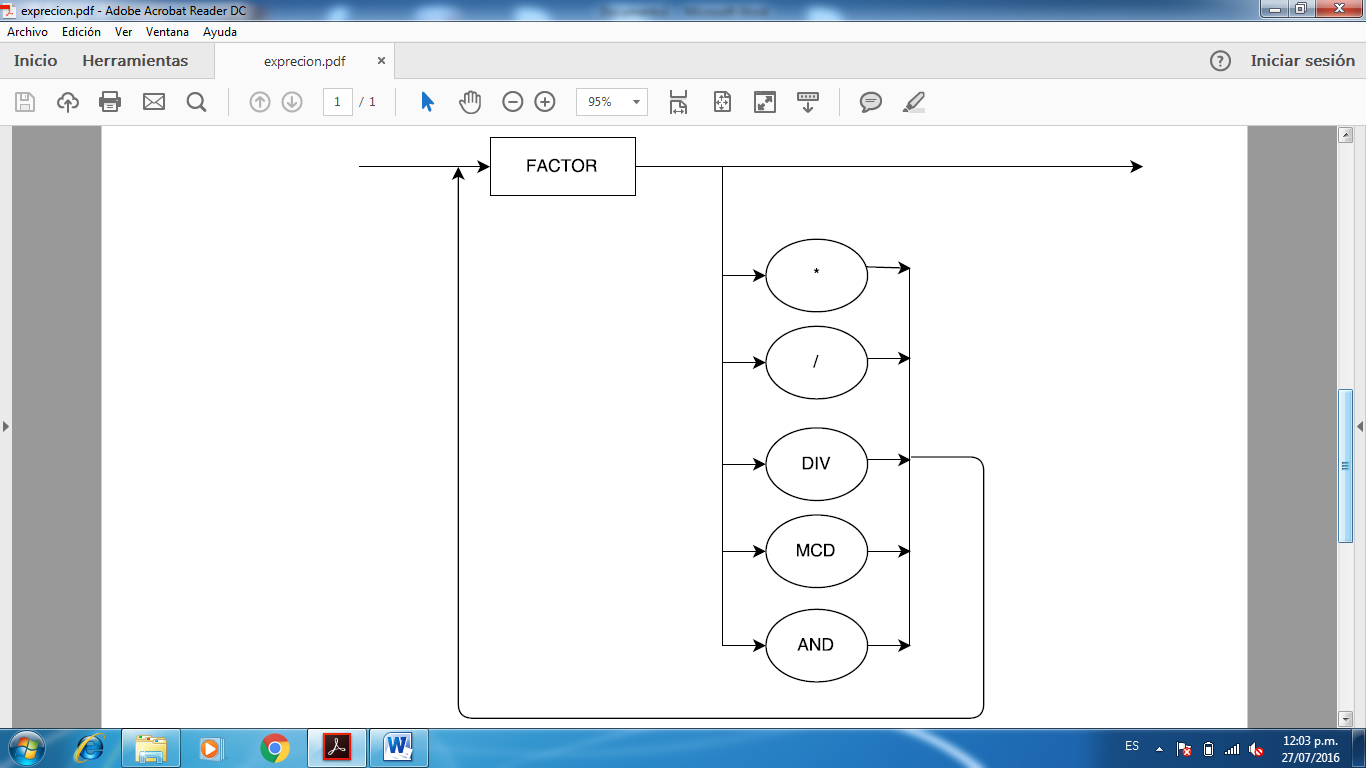
Expresión



Expresión simple

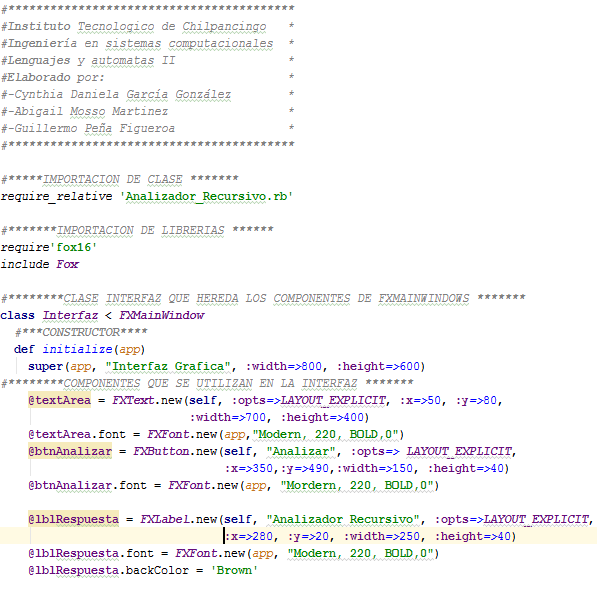


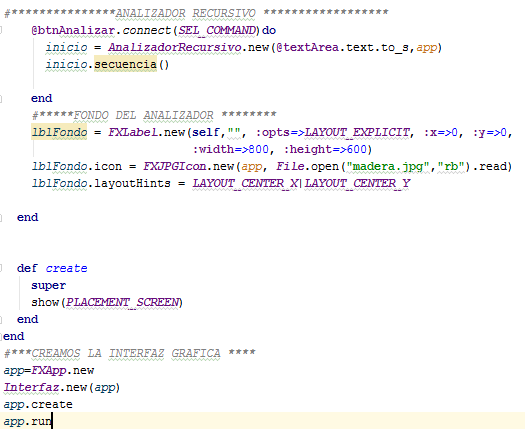
Término



Codificación en ruby:

Interfaz Gráfica:



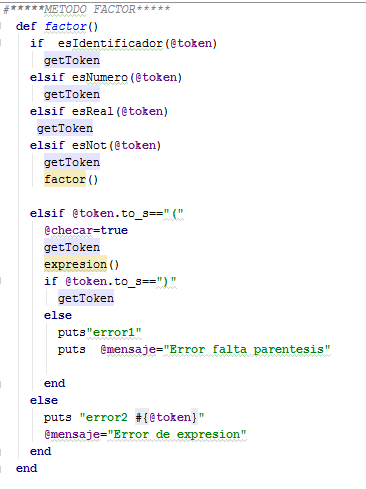


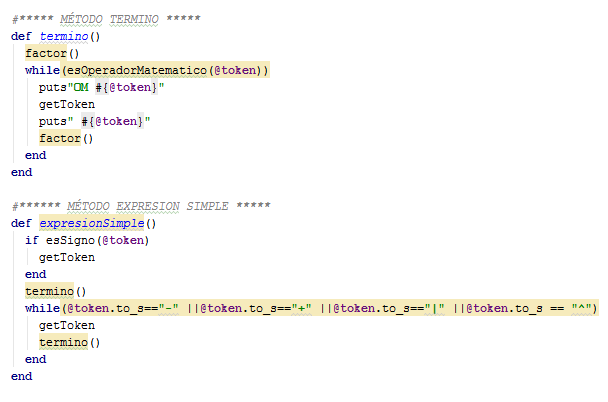
Clase Analizador léxico

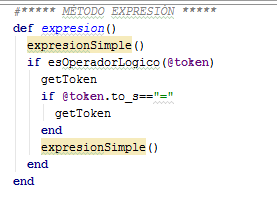


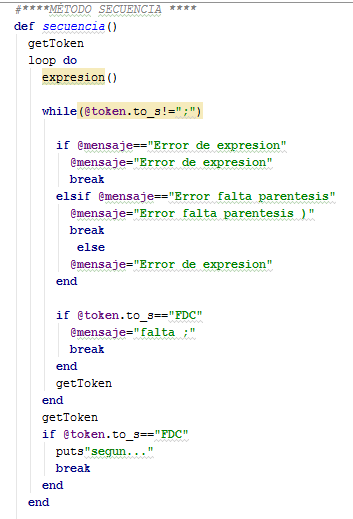


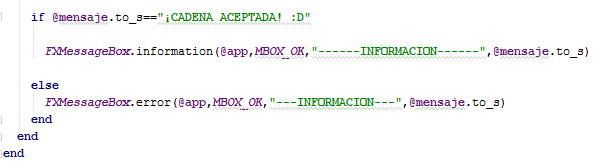






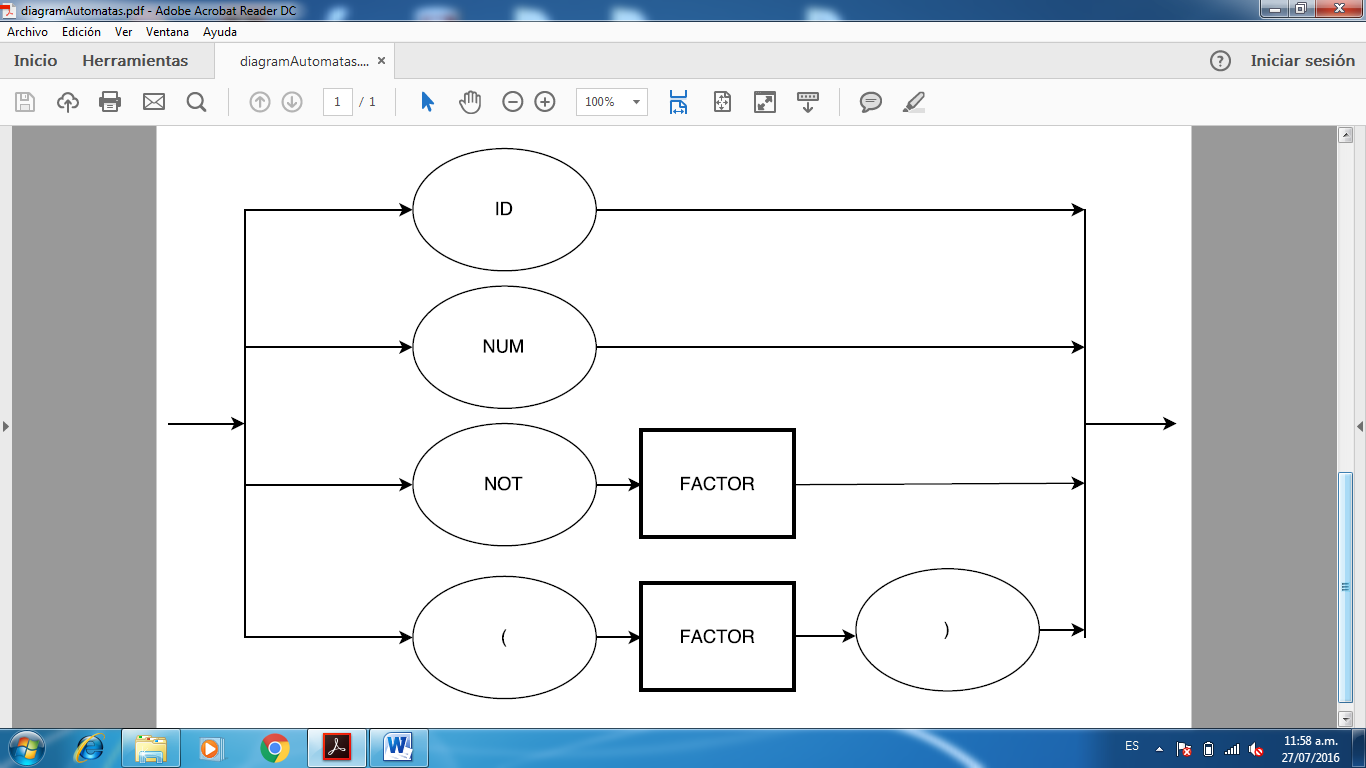






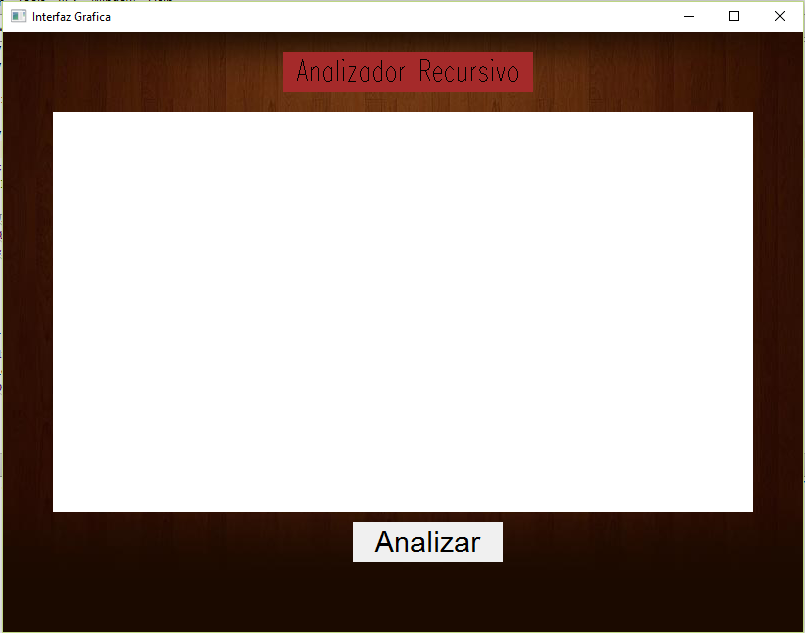
Pruebas del analizador Léxico Recursivo

1.- FACTOR

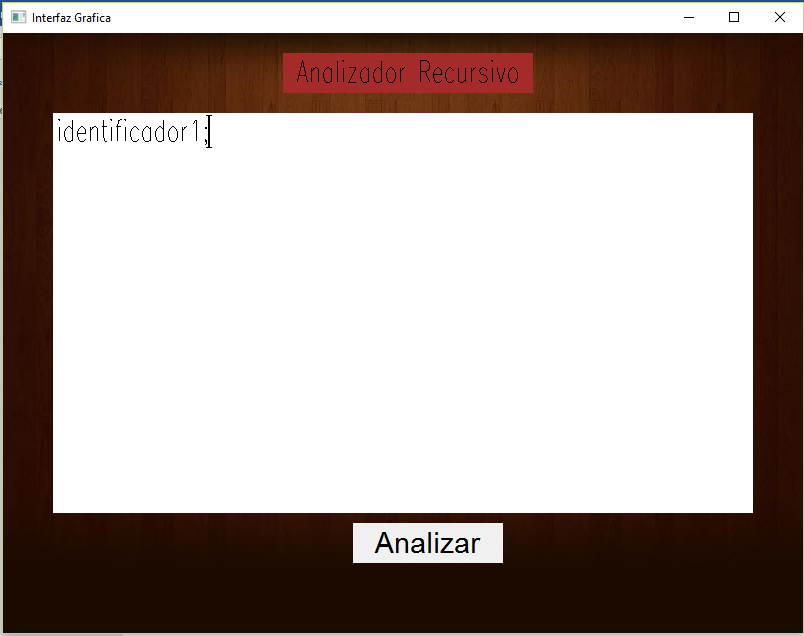


Ahora comprobaremos con el programa el diagrama de flujo del FACTOR.

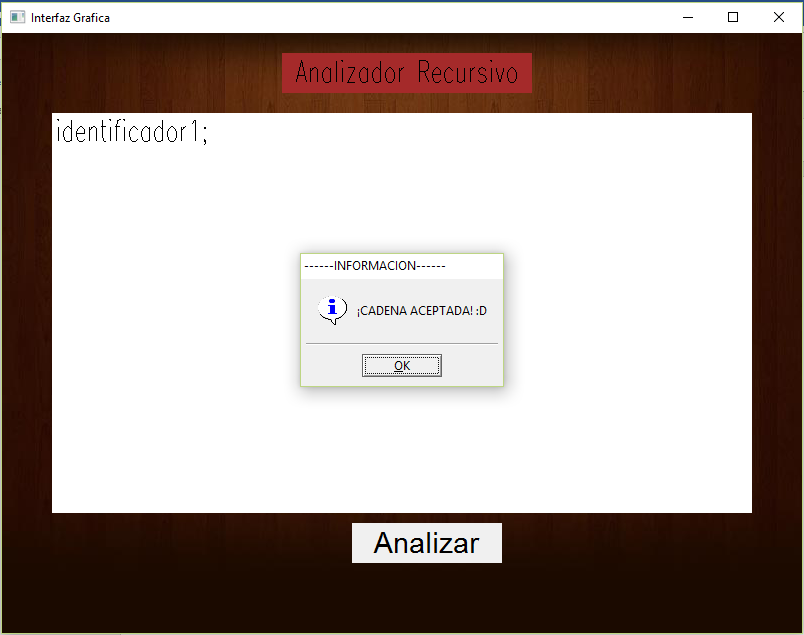
La interfaz gráfica



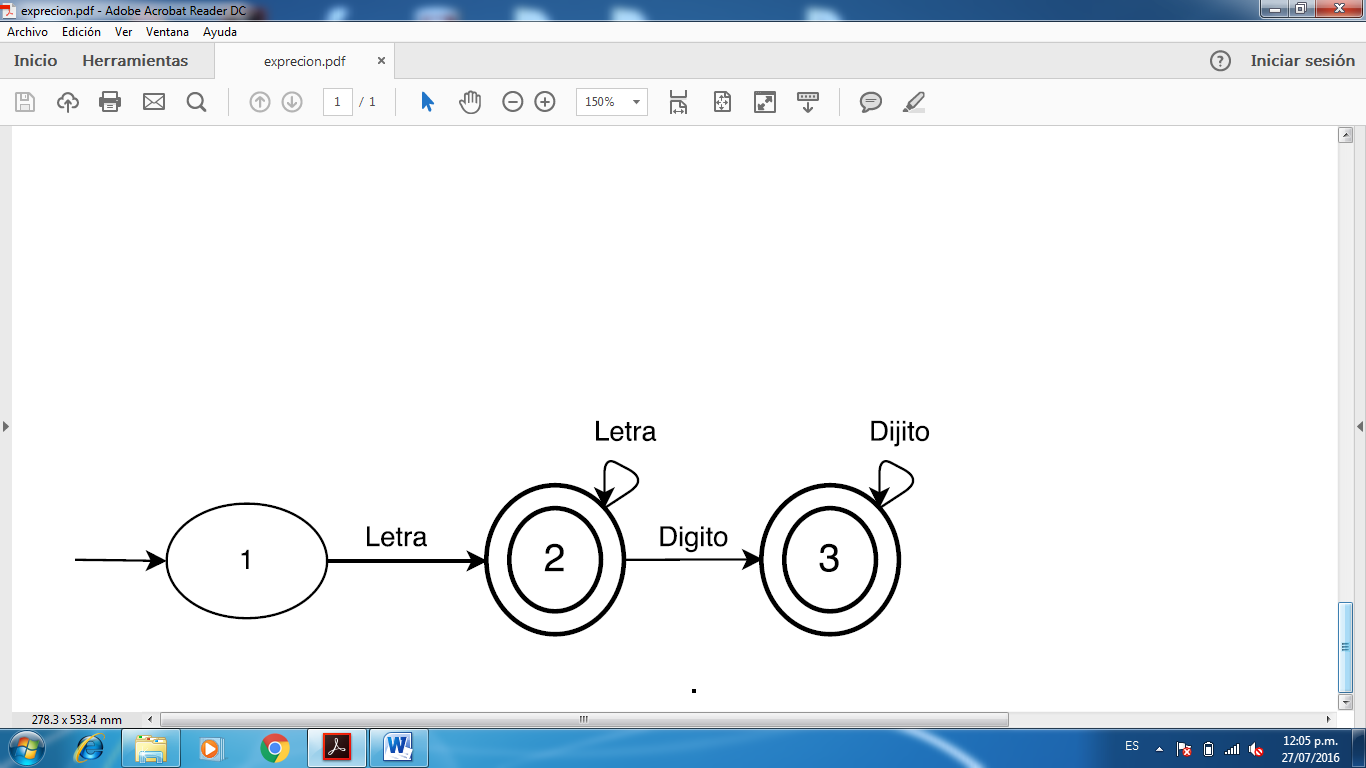
Si ingresamos un identificador:



Y damos clic en analizar:

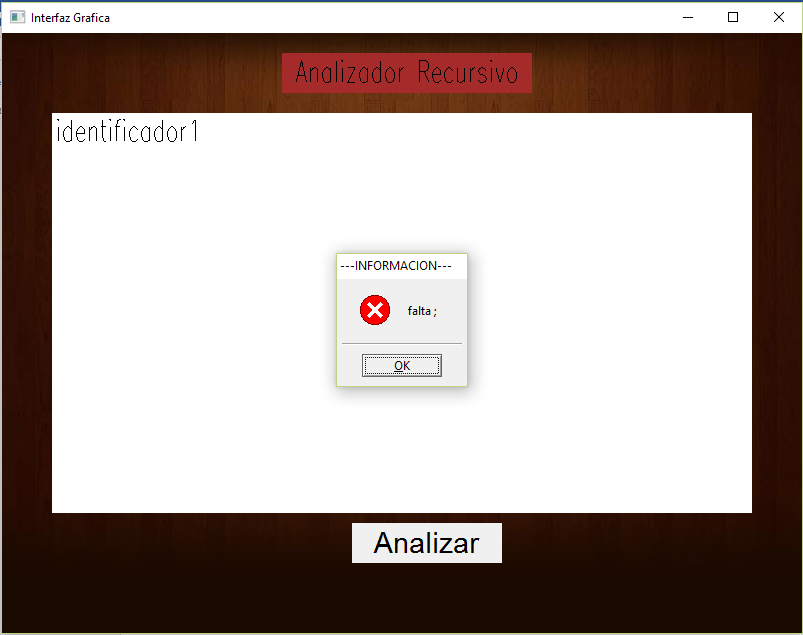


Como nos podemos dar cuenta en nuestro autómata identificador



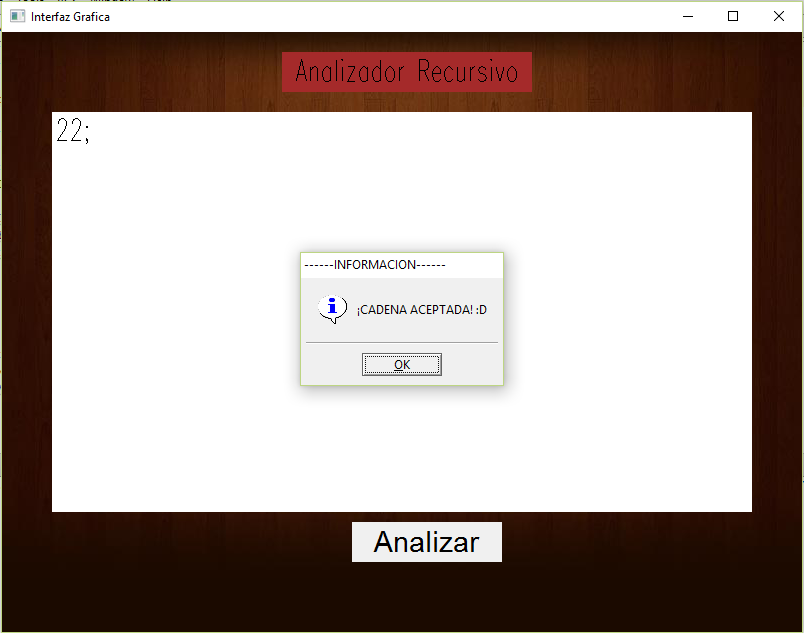
Nuestro estado inicial es 1, si es letra, nuestro estado cambiaria a estado: 2, si nuestro estado es digito cambiaria a estado: 3 y como estado 3 es final de cadena, la cadena es ACEPTADA.

Ahora para comprobar que nuestro delimitador “;” funciona lo quitaremo y analizaremos la cadena:

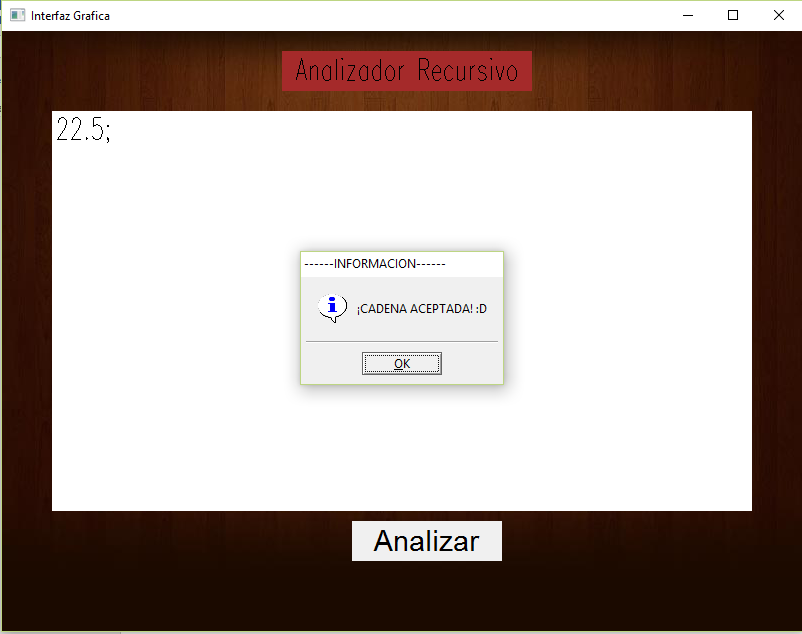


Como nos podremos dar cuenta el autómata identificador si esta correcto, pero nos indica que nos falta el delimitador “;”

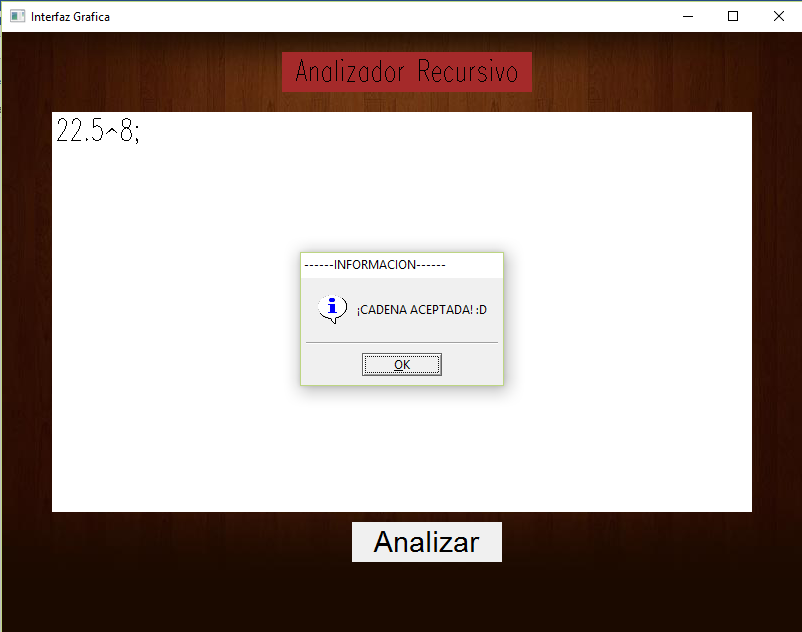
Ahora probaremos el numero



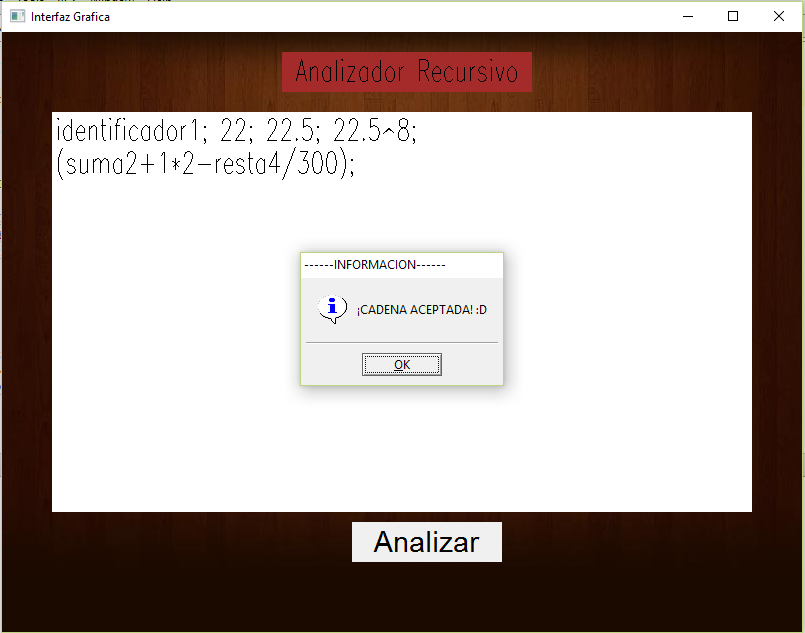
Y con número real



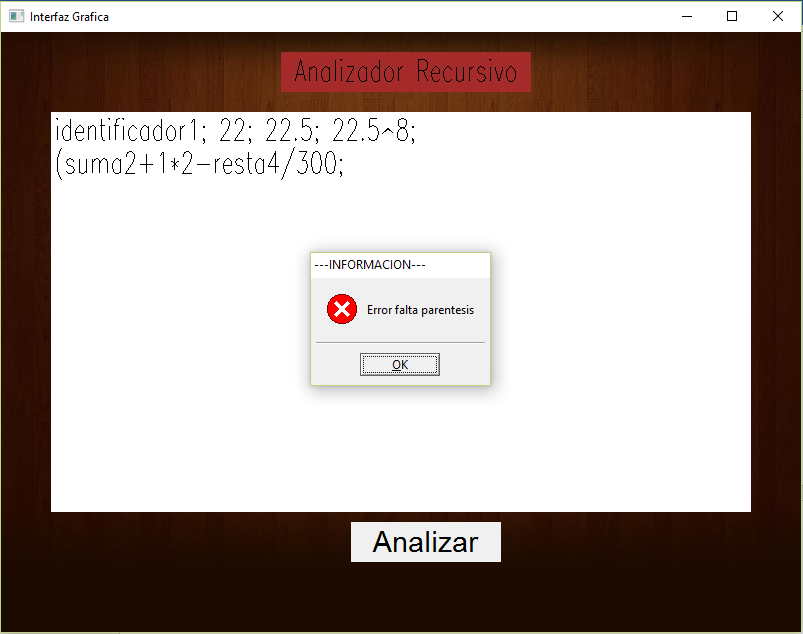
Número real con exponente



Expresion



Si le quitamos un paréntesis:



Ahora probamos todos los diagramas de flujo:

