# Proyecto 1: Analizador léxico.

Compiladores 2017-1

Elisa Viso Gurovich

September 1, 2016

## 1 Descripción

En éste proyecto se construirá el primer componente de un compilador, un analizador léxico. Para este trabajo se utilizará el generador de analizadores léxico **JFlex**. Como ya han visto en clase, un analizador léxico es aquél que recibe uno o más archivos de entrada en código fuénte y los transforma en un flujo de átomos, mismo que será posteriormente utilizado por el analizador sintáctico, su siguiente proyecto. En éste caso el lenguaje en el que estará escrito el código fuente es una versión minimizada de *python*. Los átomos presentes en la gramática del lenguaje son:

1. **IDENTIFICADOR**
2. **ENTERO**
3. **REAL**
4. **CADENA**
5. **NEWLINE**
6. **INDENT**
7. **DEDENT**

Las palabras resevadas, operador o separador también tienen que ser reconocido. Para mayor referencia revisar el archivo *atomos.pdf*

Veamos un ejemplo de cómo debe comportarse nuestro analizador léxico. Tenemos el siguiente código en python:

```
if 3 > 1:
  x = -4
  while x < 2:
    x += 1
    print x
else:
   x = "Hello world"
```

La salida esperada de nuestro analizador es:

```
IFENTEROOPERADORENTEROSEPARADORNEWLINEINDENTIDENTIFICADORSEPARADORENTERONEWLINE
WHILEIDENTIFICADOROPERADORENTEROSEPARADORNEWLINEINDENTIDENTIFICADOROPERADOR
ENTERONEWLINEPRINTIDENTIFICADORNEWLINEDEDENTDEDENTELSESEPARADORNEWLINE
INDENTIDENTIFICADORSEPARADORCADENANEWLINEDEDENT
```

A los ojos del analizador sintáctico, la salida es suficientemente entendible de ésta manera. Pero para nosotros sería mejor de ésta manera:

```
KEYWORD(if)INTEGER(3)OPERATOR(>)INTEGER(1)SEPARADOR(:)NEWLINE
INDENT(2)IDENTIFICADOR(x)OPERATOR(=)OPERATOR(-)INTEGER(4)NEWLINE
KEYWORD(while)IDENTIFICADOR(x)OPERATOR(<)INTEGER(2)SEPARADOR(:)NEWLINE
INDENT(4)IDENTIFICADOR(x)OPERATOR(+=)INTEGER(1)NEWLINE
KEYWORD(print)IDENTIFICADOR(x)NEWLINE
DEDENT
DEDENT
KEYWORD(else)SEPARADOR(:)NEWLINE
INDENT(3)IDENTIFICADOR(x)OPERATOR(=)CADENA(Hello world)NEWLINE
DEDENT
```

Cabe destacar que los espacios y saltos de línea son únicamente para hacer mas legible la salida, no son átomos que deba producir el analizador léxico. Sólo para facilitar la lectura, salidas como la anterior, con espacios y saltos de línea, son las que se esperan para éste proyecto. Ahora veremos cómo se comporta bajo el siguiente error léxico:

```
if 0 < 1:
    x = 2 + 9
  print x
```

Salida:

```
KEYWORD(if)ENTERO(0)OPERADOR(<)ENTERO(1)SEPARADOR(:)NEWLINE
INDENT(2)IDENTIFICADOR(x)OPERADOR(=)ENTERO(2)OPERADOR(+)ENTERO(9)NEWLINE
Error de indentacion, linea 3
```

Veamos ahora el siguiente ejemplo:

```
if 3 < 1:
   print 2
      print 7
```

Salida:

```
KEYWORD(if)ENTERO(3)OPERADOR(<)ENTERO(1)SEPARADOR(:)NEWLINE
INDENT(3)KEYWORD(print)ENTERO(2)NEWLINE
INDENT(6)KEYWORD(print)ENTERO(7)NEWLINE
DEDENT
DEDENT
```

Entre las dos instrucciones de print no hay una que nos indique la creación de un nuevo bloque, por lo que no es un programa válido en python, sin embargo, el analizador léxico no es el encargado de detectar éste tipo de errores. La regla general es que el analizador léxico debe detectar átomos mal formados, mientras que el analizador sintáctico secuencias de átomos mal formadas respecto a la sintaxis del lenguaje.

La salida puede ser devuelta en algún archivo o a la salida estándar, pero en ambos casos debe reportar errores con número de línea.

## 2 Ejercicios para el laboratorio (4pts):

1. Investigar en que consiste la funcionalidad de contextos que provee JFlex
2. ¿Cómo uso la funcionalidad de depuración que provee JFlex?
3. Empezar el analizador léxico. Reconocer los átomos de **DEDENT**, **INDENT** y **NEWLINE**.
4. En el uso del lenguaje, ¿la indentación significativa facilita echar código?.
   En la implementación, ¿la indentación significativa facilita el reconocimiento de bloques?

## 3 Ejercicios para llevar(5pts):

1. Terminar de reconocer los átomos

## 4 Administrativos(1pts):

1. El proyecto deberá se entregado antes del 11.09.16.
2. El código deberá estar comentado.
3. Estará en /Proyectos/Proyecto1
4. Los archivos que deberán estar en la carpeta anterior deben ser:
   - Flex.flex: archivo de entrada para JFlex.
   - Readme.md: describir los pasos para probarlo.
   - makefile.
5. En caso de que tengan que subir script de lo harán antes del 9.11.16 y se subirá al grupo asociado a este curso junto con la salida esperada. No más largo de 50 líneas. `https://groups.google.com/a/ciencias.unam.mx/forum/?hl=es-419#!forum/compiladores2017-1`

## 5 Puntos Extras:

1. Las cadenas pueden contener comillas simples y dobles siempre y cuando estén escapadas.
2. Se detecta un error de sintáxis cuando las primeras líneas tienen indentación diferente de 0. Es decir:

```
   print 2
print 2
```

salida:

```
Error de indentación. Línea 1.
```