**Tarea 2**

**Lex&Yacc**

Es un programa hecho para el proceso léxico de “character input streams”, acepta problemas de alto nivel, orientados específicamente para concordancia de strings y crea un programa general que reconoce el lenguaje de expresiones regulares.

El formato general es: {definiciones} %% {reglas} %% {subrutinas} donde las definiciones y las subrutinas normalmente son omitidas. El segundo “%%” es opcional, pero el primero es requerido ya que marca el inicio de las reglas.

**CocoR**

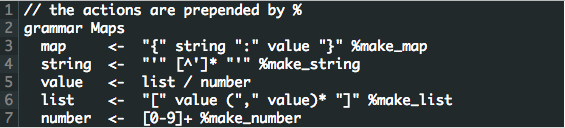
Es un generador de compilación, que toma los atributos gramaticales de un lenguaje, el cual genera un scanner y un “parser” para este. Este scanner funciona como un autómata finito determinístico, el cual usa recursividad para su mejor funcionalidad.

**Canopy**

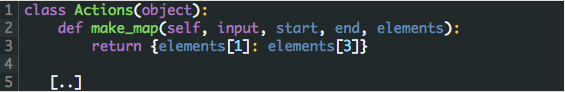
Es un “parser” compilador que ataca principalmente a Java, Javascript, Python y Ruby. La forma en que funciona es, toma un archivo describiendo un “parsing” gramatical y lo compila en un modulo de “parser” en el lenguaje escrito.

La gramática Canopy tiene las características de usar “action annotation” para el código propio en el parser; en otras palabras solo se tiene que escribir el nombre de una función a lado de una regla, después solo se implementa la función en su código fuente.

Gramatica Canopy con acciones:



Archivo Python con el codigo de accion:



**Bibliografia**

* http://dinosaur.compilertools.net/
* http://dinosaur.compilertools.net/lex/index.html
* http://www.ssw.uni-linz.ac.at/Research/Projects/Coco/
* https://tomassetti.me/parsing-in-python/#arpeggio