Universidad Católica San Pablo

Compiladores

Práctica 4 – Analizadores sintáctico descendente recursivo predictivo (1era parte)

Objetivo:

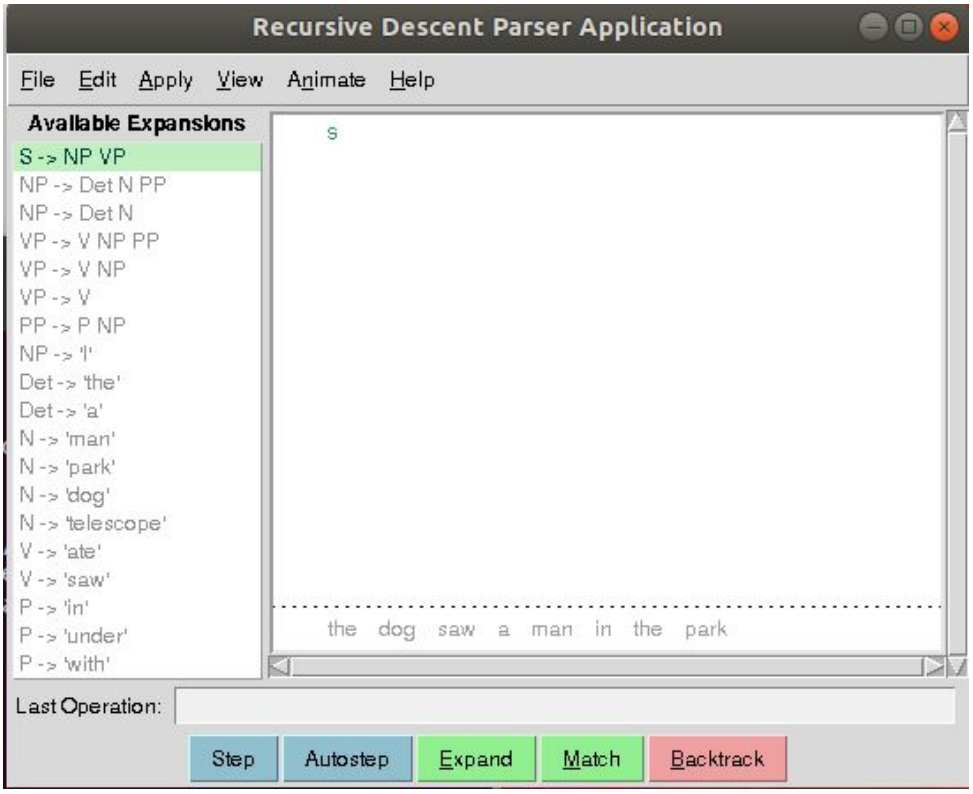
- Construir un analizador sintáctico descendente predictivo.
- Modelar una gramática CFG.
- Escribir una tabla de análisis sintáctico
- Escribir el algoritmo de reconocimiento de una cadena utilizando nuestro analizador

Preliminares:

- Probando un analizador sintáctico descendente: https://www.nltk.org/book/ch08.html
- Instalar nltk (**`pip install nltk`**)
- Ejecutar:

```
>>> import nltk
>>> nltk.app.rdparser()
```

- Divertirse con la herramienta ^_^
-



Universidad Católica San Pablo

Indicaciones:

1. **Estableciendo una gramática**

   Especificar los constructores y sobrecarga de operadores para las siguientes clases:

```
class Produccion:
  izq  #Componente de la izquierda
  der  #Componente(s) de la derecha

class Gramatica:
    produccion = []    #Lista de producciones
    terminales = {}    #Conjunto de terminales
    noterminales = {} #no terminales

    def print(self):  #Crear una función para imprimir
       pass
```

2. **Crear métodos para cargar la gramática a partir de un texto y para obtener una producción.**

```
class Gramatica:
   def cargar(self,texto):
      #cargar un texto en gramatica

   def getProduccion(self,izq):
      #retornar el componente(s) de la derecha de la
      #produccion apuntada por izq
```

Universidad Católica San Pablo

3. Gramática a modelar:

```
E  := T Ep
Ep := + T Ep
Ep := - T Ep
Ep := lambda
T  := F Tp
Tp := * F Tp  |  /  F Tp  | lambda
F  := ( E ) | num | id
```

Implementar el método para imprimir la gramática

4. Tabla sintáctica de predicción:

	+	-	*	/	(	)	num	id	$
E					T Ep		T Ep	T Ep	
Ep	+ T Ep	- T Ep				lambda			lambda
T					F Tp		F Tp	F Tp	
Tp	lambda	lambda	* F Tp	/ F Tp		lambda			lambda
F					(E)		num	id	

Diseñar su solución y crear la tabla de forma estática

```
class TAS:
   tablaSintactica # mapea NoTerminales vs Terminales  => produccion
   def print():
      pass
```

y para llenar estáticamente

```
   def llenarEstaticamente(self):
      self.tablaSintactica[“Ep”][“+”] = [“+”,”T”, “Ep”]    #una forma
      self.tablaSintatica.insertar( “E”, “(“, [“T”,“Ep”] )  #otra forma
      self.tablaSintactica.insertar(“E”,”(“, Produccion(“E”, [“T”,“Ep”))  #más otra
      #u otra forma que usted desee
```