Universidad Simón Bolívar
Departamento de Computación y Tecnología de la Información
Traductores e Interpretadores
Septiembre-Diciembre 2018

# Etapa II: Análisis Sintáctico con Árbol Sintáctico Abstracto (AST)

En esta entrega usted deberá diseñar una gramática libre de contexto para el lenguaje PascalUSB y utilizarla para implementar un reconocedor. Su reconocedor debe además construir el árbol sintáctico abstracto del programa reconocido e imprimirlo de forma legible por pantalla.

Su gramática debe ser lo suficientemente completa tal que pueda reconocer cada instrucción y expresión de PascalUSB , y combinarlos para hacer programas. Note que su gramática solo debe chequear la forma de la entrada, pero no su significado; es decir, no es necesario que detecte problemas de tipos o variables no declaradas.

Para construir el árbol sintáctico debe implementar las clases necesarias para representar las instrucciones y expresiones de PascalUSB.

## 1. Formato

Su programa principal deberá llamarse `pascalusb` y recibirá como único argumento el nombre del archivo a analizar. La ejecución del mismo mostrará en pantalla la representación del árbol de sintaxis (descrita más adelante).

## 2. Errores

Si el analizador lexicográfico consigue errores, deben reportarse todos y abortar la ejecución de la misma manera que se realizó en la primera entrega.

Basta que su analizador sintáctico aborte ante el primer error de sintáxis. Si bien las herramientas en uso le permitirían implantar una recuperación de errores sintácticos completa, no es el objetivo de este curso, así que no invierta tiempo en ello. Sin embargo, al encontrar el primer error sintáctico, el reconocedor debe abortar la ejecución indicando la línea y columna del error, así como el token que causó el problema. Por ejemplo:

```
program
  begin
    a := ;
    b := ;
  end
```

Al pasar este archivo por su programa debe obtenerse un resultado similar a:

```
$ Sintax error in row 3, column 10: unexpected token ';'.
```

## 3. Impresión de AST

Para la impresión del árbol sintáctico considere el siguiente ejemplo de código en PascalUSB:

```
program
begin
  declare
     a, b, i as int;
     x, y as inter

  // iterar sobre el primer intervalo
  read a;
  read b;
  x := a..b;
  for i in x do
    println "Variable \"i\" es igual a: ", i;

  // iterar sobre el segundo intervalo
  read y;
  for j in y do
    print j, ", ";

  a := 3 + b;
  b := -4;

  case b of
   x ==> println b
   y ==> println a
   a..b ==> begin
              declare z as inter
              println a, b;
              z := x <> y;
              println min(z), "..", max(z)
            end
  end;

  i := 3;
  while i < 10 do
     begin
      read i;
      print "Still here!"
     end
end
```

El resultado del analizador al leer el programa anterior, debe devolver impreso el árbol AST con el siguiente formato

```
Block
 Declare
  a, b, i as int
  Sequencing
   x, y as inter
 Read
  Ident: a
 Sequencing
```

```
Read
 Ident: b
Sequencing
 Asig
  Ident: x
  Exp
   SoForth
    Ident: a
    Ident: b
 Sequencing
  For
   In
    Ident: i
    Exp
     Ident: x
   Println
    "Variable \"i\" es igual a: "
    Ident: i
  Sequencing
   Read
    Ident: y
   Sequencing
    For
     In
      Ident: j
      Exp
       Ident: y
     Print
      Ident: j
      ", "
    Sequencing
     Asig
      Ident: a
      Exp
       Plus
        Literal: 3
        Ident: b
     Sequencing
      Asig
       Ident: b
       Exp
        Literal: -4
      Sequencing
       Case
        Exp
         Ident: b
        Guard
         Exp
          Ident: x
         Println
          Ident: b
        Guard
         Exp
          Ident: y
```

```
 Println
  Ident: a
Guard
 Exp
  SoForth
   Ident: a
   Ident: b
 Block
  Declare
   z as inter
  Println
   Ident: a
   Ident: b
  Sequencing
   Asig
    Ident: z
    Exp
     Cap
      Ident: x
      Ident: y
   Sequencing
    Println
     Min
      Ident: z
     ".."
     Max
      Ident: z
Sequencing
 Asig
  Ident: i
  Exp
   Literal: 3
 Sequencing
  While
   Exp
    Less
     Ident: i
     Literal: 10
   Block
    Read
     Ident: i
    Print
     "Still here!"
```

# 4. Lenguajes y Herramientas a usar

Para la implementación de este proyecto se permitirá el uso de las siguientes herramientas. Estos son:

- Python:
  - Interpretador *python*
  - Generador de analizadores lexicográficos y sintácticos PLY
- Ruby:

- Interpretador *ruby*
- Generador de analizadores sintácticos *Racc*

# 5. Entrega de la Implementación

La entrega del proyecto es el domingo de la semana 6 (4 de Noviembre) por Aula Virtual. Su entrega debe incluir lo siguiente:

- Un archivo llamado `gramatica.txt` que contenga la gramática propuesta por usted para reconocer PascalUSB . Esta gramática debe coincidir con la implementada por su reconocedor y debe seguir el siguiente formato:

```
S -> A B C
   | A B

A -> C
...
```

  indicando cual es el símbolo inicial de la gramática.

- Un archivo comprimido `tar.gz` con el código fuente de su proyecto, debidamente documentado y el archivo `gramatica.txt`, colocado en el Aula Virtual. El nombre del archivo deber ser **Etapa2-XX-YY.tar.gz** donde **XX-YY** son los carné de los integrantes del grupo.

El no cumplimento de los requerimientos podría resultar en el rechazo de su entrega.

---

Federico Flaviani / federico.flaviani@gmail.com / Octubre 2018