Desarrollo de analizadores léxicos

Procesadores de Lenguaje - GII (UCM)

Grupo 18

Jorge Ortega, Alejandro Tobías

Miguel Amato y Daniela Vidal

Índice

1. Desarrollo manual de un analizador léxico para Tiny(0)

1.1. Enumeración de las clases léxicas

Las clases léxicas definen los elementos básicos que pueden ser reconocidos por el analizador léxico de Tiny(0):

Clases Univaluadas:

- Una clase léxica por cada palabra reservada:
  true, false, int, real, bool, and, or, not
- Una clase léxica para cada operador aritmético:
  +, -, *, /
- Una clase léxica para cada operador relacional:
  <, >, <=, >=, ==, !=
- Una clase para operador de asignación:
  =
- Una clase léxica para cada símbolo de puntuación:
  \(\) \; \{ \} \&& \@

Clases Multivaluadas:

- Identificador/Variables: comienzan necesariamente por una letra o subrayado (_), seguida de una secuencia de cero o más letras, dígitos, o subrayado.
- Literal entero: comienzan, opcionalmente, con un signo + o -. Seguidamente debe aparecer una secuencia de 1 o más dígitos (no se admiten ceros no significativos a la izquierda).
- Literal real: comienzan, obligatoriamente, con una parte entera, cuya estructura es como la de los números enteros, seguida de bien una parte decimal, bien una parte exponencial, o bien una parte decimal seguida de una parte exponencial. La parte decimal comienza con un '.', seguido de una secuencia de 1 o más dígitos (no se permite la aparición de ceros no significativos a la derecha). Por último, y también opcionalmente, puede aparecer una parte exponencial (e o E, seguida de un exponente, cuya estructura es igual que la de los números enteros).

Cadenas Ignorables:

- Espacios en blanco: blanco, tabulador, retorno de carro, retroceso, salto de línea y EOF.
- Comentarios: Son comentarios de una línea. Comienzan con ##, seguida de una secuencia de 0 o más caracteres, a excepción del salto de línea y el EOF.

1.2. Especificación formal del léxico

Definiciones auxiliares:

- letra = ((a-z) | (A-Z) | _)
- digitoPositivo = [1-9]
- digito = {digitoPositivo} | 0
- parteEntera = ({digitoPositivo} {digito}* | 0)
- parteDecimal = ({digito}* {digitoPositivo} | 0)
- decimal = \.{parteDecimal}
- exponencial = (e|E)[\+,\-]?{parteEntera}
- and = (a | A)(n | N)(d | D)
- or = (o | O)(r | R)
- not = (n | N)(o | O)(t | T)
- bool = (b | B)(o | O)(o | O)(l | L)
- int = (i | I)(n | N)(t | T)
- real = (r | R)(e | E)(a | A)(l | L)
- false = (f | F)(a | A)(l | L)(s | S)(e | E)
- true = (t | T)(r | R)(u | U)(e | E)

Definiciones léxicas:

- identificador = {letra}({letra}|{digito})*
- literalEntero = [\+,\-]?{parteEntera}
- literalReal = ({literalEntero}{decimal})|({literalEntero}{exponencial})|({literalEntero}{decimal}{exponencial})
- suma = \+
- resta = \-
- mul = *
- div = \/

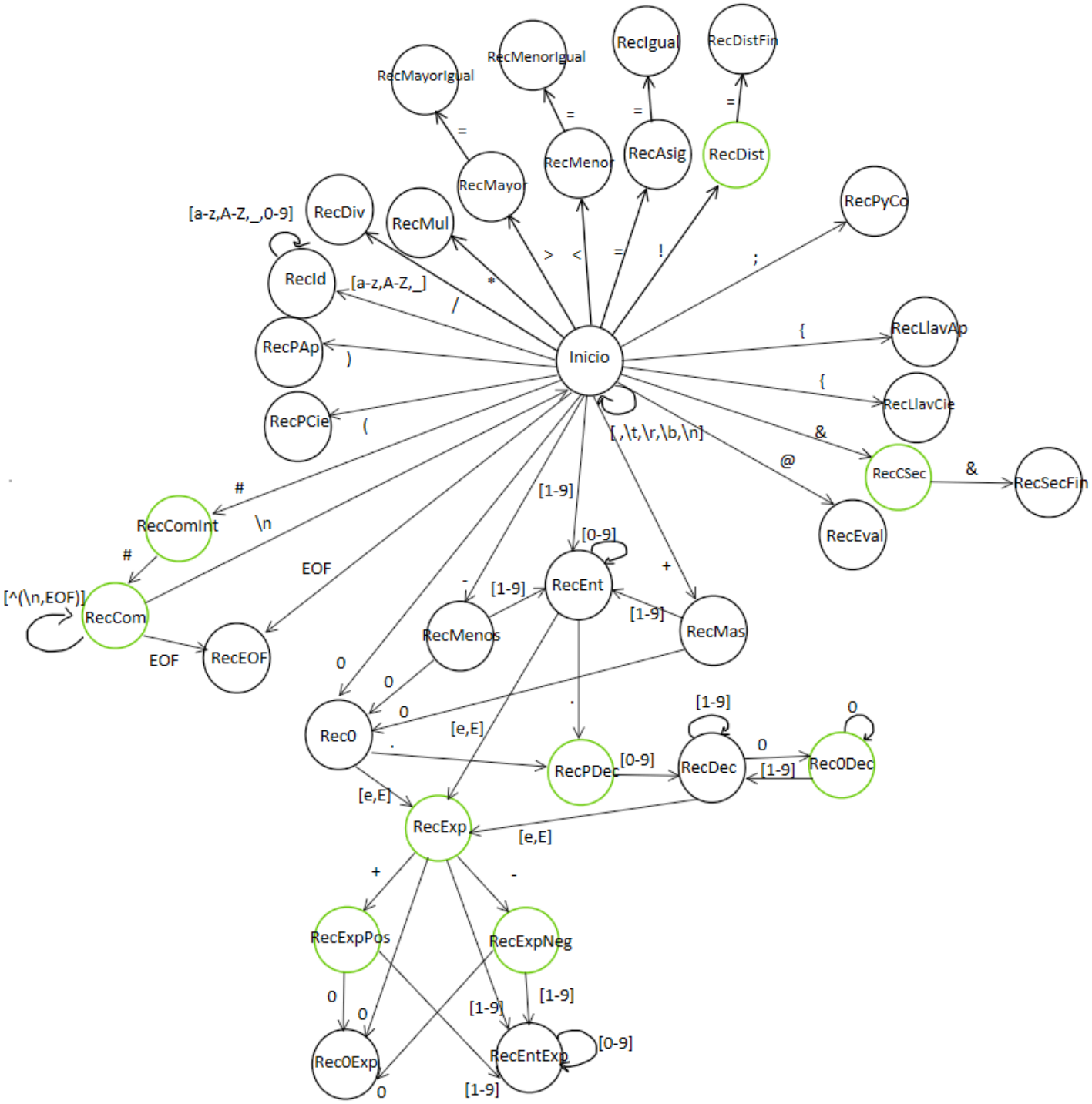
- and = {and}
- or = {or}
- not = {not}
- menor = \<
- mayor = \>
- menorIgual = \<\=
- mayorIgual = \>\=
- igualdad = \=\=
- distinto = \!\=
- parAp = \(
- parCierre = \)
- asignación = \=
- pyComa = \;
- llaveAp = \{
- llaveCierre = \}
- cambioSec = \&\&
- eval = \@
- true = {true}
- false = {false}
- int = {int}
- real = {real}
- bool = {bool}

Definiciones de cadenas ignorables:

- separador = [, \t, \r, \b, \n]
- comentario = ##([^\n,EOF])*

1.3. Diseño de un analizador léxico

(Verde: estado no final | Negro: estado final)



2. Desarrollo manual de un analizador completo para Tiny

2.1. Enumeración de las clases léxicas

Las clases léxicas definen los elementos básicos que pueden ser reconocidos por el analizador léxico de Tiny:

Clases Univaluadas:

- Una clase léxica por cada palabra reservada:
  int, real, bool, string, and, or, not, null, true, false, proc, if, else, while, struct, new, delete, read, write, nl, type, call.
- Una clase léxica para cada operador aritmético:
  +, -, *, / , %
- Una clase léxica para cada operador relacional:
  <, >, <=, >=, ==, !=
- Una clase para operador de asignación:
  =
- Una clase léxica para cada símbolo de puntuación:
  (,), ;, [,], ., ^, ,, {, }, &, &&, @.

Clases Multivaluadas:

- Identificador/Variables: Comienzan necesariamente por una letra o subrayado (_), seguida de una secuencia de cero o más letras, dígitos, o subrayado (_) .
- Literal entero: comienzan, opcionalmente, con un signo + o -. seguidamente debe aparecer una secuencia de 1 o más dígitos (no se admiten ceros no significativos a la izquierda).
- Literal real: comienzan, obligatoriamente, con una parte entera, cuya estructura es como la de los números enteros, seguida de bien una parte decimal, bien una parte exponencial, o bien una parte decimal seguida de una parte exponencial. La parte decimal comienza con un '.', seguido de una secuencia de 1 o más dígitos (no se permite la aparición de ceros no significativos a la derecha). Por último, y también opcionalmente, puede aparecer una parte exponencial (e o E, seguida de un exponente, cuya estructura es igual que la de los números enteros).

- Literal cadena: comienzan con comilla doble ("), seguida de una secuencia de 0 o más caracteres distintos de ", seguida de ". Los caracteres pueden incluir las siguientes secuencias de escape: retroceso (\b), retorno de carro (\r), tabulador (\t), y salto de línea (\n).

Cadenas Ignorables:

- Espacios en blanco: blanco, tabulador, retorno de carro, salto de línea.
- Comentarios: comienzan por ##. Son comentarios de una línea.

2.2. Especificación formal del léxico

Definiciones auxiliares:

- letra = ((a-z) | (A-Z) | _)
- digitoPositivo = [1-9]
- digito = {digitoPositivo} | 0
- parteEntera = ({digitoPositivo} {digito}* | 0)
- parteDecimal = ({digito}* {digitoPositivo} | 0)
- decimal = \.{parteDecimal}
- exponencial = (e|E)[\+,\-]?{parteEntera}
- and = (a | A)(n | N)(d | D)
- or = (o | O)(r | R)
- not = (n | N)(o | O)(t | T)
- int = (i | I)(n | N)(t | T)
- real = (r | R)(e | E)(a | A)(l | L)
- bool = (b | B)(o | O)(o | O)(l | L)
- string = (s | S)(t | T)(r | R)(i | I)(n | N)(g | G)
- null = (n | N)(u | U)(l | L)(l | L)
- false = (f | F)(a | A)(l | L)(s | S)(e | E)
- true = (t | T)(r | R)(u | U)(e | E)
- proc = (p | P)(r | R)(o | O)(c | C)
- if = (i | I)(f | F)
- else = (e | E)(l | L)(s | S)(e | E)
- while = (w | W)(h | H)(i | I)(l | L)(e | E)
- struct = (s | S)(t | T)(r | R)(u | U)(c | C)(t | T)
- new = (n | N)(e | E)(w | W)

- delete = (d | D)(e | E)(l | L)(e | E)(t | T)(e | E)
- read = (r | R)(e | E)(a | A)(d | D)
- write = (w | W)(r | R)(i | I)(t | T)(e | E)
- nl = (n | N)(l | L)
- type = (t | T)(y | Y)(p | P)(e | E)
- call = (c | C)(a | A)(l | L)(l | L)

Definiciones léxicas:

- identificador = {letra}({letra}|{digito})*
- literalEntero = [\+,\-]?{parteEntera}
- literalCadena = \"[^\"]*\"
- literalReal = ({literalEntero}{decimal})|({literalEntero}{exponencial})|({literalEntero}{decimal}{exponencial})
- suma = \+
- resta = \-
- mul = *
- div = \/
- mod = \%
- and = {and}
- or = {or}
- not = {not}
- menos = \(-)
- parAp = \(
- parCierre = \)
- menor = \<
- mayor = \>
- menorIgual = \<\=
- mayorIgual = \>\=
- igualdad = \=\=
- distinto = \!\=
- asignación = \=
- coma = \,
- pyComa = \;

- corAp = \[
- corCierre = \]
- punto = \.
- puntero = \^
- llaveAp = \{
- llaveCierre = \}
- referencia = \&
- cambioSec = \&\&
- eval = \@
- int = {int}
- real = {real}
- bool = {bool}
- string = {string}
- null = {null}
- true = {true}
- false = {false}
- proc = {proc}
- if = {if}
- else = {else}
- while = {while}
- struct = {struct}
- new = {new}
- delete = {delete}
- read = {read}
- write = {write}
- nl = {nl}
- type = {type}
- call = {call}

Definiciones de cadenas ignorables:

- separador = [, \t, \r, \b, \n]
- comentario = ##([^\n,EOF])*