TINY 1

Grupo 13
Gorka Silva Ramón
Sofía Capmany Fernández

ÍNDICE DE CONTENIDOS

1. ANALIZADOR LÉXICO

1.1 Clases Léxicas

I. Clases léxicas univaluadas

&& separador de declaraciones e instrucciones
; punto y coma
= operador "=" para las instrucciones de asignación.
+ operador de suma.
- operador de resta.
***** operador de multiplicación.
/ operador de división.
% operador binario módulo.
< operador relacional menor que (Be Lower Than)
> operador relacional mayor que (Be Grater Than)
<= operador relacional menor o igual (Be Lower or Equal)
>= operador relacional mayor o igual (Be Grater or Equal)
== operador relacional "==" (Be Equal)
!= operador relacional "!=" (Be Not Equal)
(paréntesis de apertura.
) paréntesis de cierre.
[corchete de apertura.
] corchete de cierre.
{ llave de apertura.
} llave de cierre.
-> operador acceso a campo de variable.
, separador parámetros función.
. operador acceso a campo de variable.
& Dirección de la variable (paso por variable).

— Palabras reservadas:

int nombre de tipo de variable entera
real nombre de tipo de variable real
bool nombre de tipo de variable booleana
true valor verdadero de las variables booleanas.
false valor falso de las variables booleanas.
and operador lógico and.
or operador lógico or.
not operador lógico not.
string nombre de tipo de variable literal cadena.
null expresión básica de tipo null.
proc palabra reservada para la declaración de un procedimiento.
if palabra reservada.
then palabra reservada.
else palabra reservada.
endif palabra reservada.

while palabra reservada.
do palabra reservada.
endwhile palabra reservada.
call instrucción invocación a procedimiento.
recod tipo registro.
array tipo array.
of instrucción declaración tipo.
pointer tipo puntero.
new instrucción de reserva de memoria.
del instrucción de liberación de memoria.
read instrucción de lectura.
write instrucción de escritura.
nl instrucción de nueva línea.
var declaración de variables.
type palabra reservada para declaraciónes de tipo.

II: Clases léxicas multivaluadas

Identificadores nombre de una variable. Comienzan necesariamente por una letra, seguida de una secuencia de cero o más letras, dígitos, o subrayado (_)

Literales enteros literal numérico 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).

Literales reales literal numérico 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).

Literales cadena literal cadena. Comienzan con comilla doble ("), seguida de una secuencia de 0 o más caracteres distintos de ", retroceso (\b), retorno de carro (\r), y salto de línea (\n), seguida de ".

III: Cadenas ignorables

Separadores posible separación entre tokens, puede ser un espacio, un tabulador (\t) o un salto de línea (\n), un retroceso (\b) o un retorno de carro (\r).
‘ ‘ espacio (Space Bar)
\t tabulador
\n salto de línea (New Line)

\b	restroceso (Back Space)
\r	retorno de carro (Carrier return)

Comentarios comentarios de línea. Comienzan por #, seguida de una secuencia de 0 o más caracteres, a excepción del salto de línea.

1.2 Definiciones regulares

I. Clases léxicas univaluadas

SEP_PROG ≡ **&&**
PTO_COMA ≡ **;**
ASIG ≡ **=**
MAS ≡ **\+**
MENOS ≡ **\-**
POR ≡ *****
DIV ≡ **/**
MOD = **%**
BLT ≡ **<**
BGT ≡ **>**
BLE ≡ **<=**
BGE ≡ **>=**
BEQ ≡ **==**
BNE ≡ **!=**
PAP ≡ **\(**
PCIERRE ≡ **\)**
CAP = **\[**
CCIERRE = **\]**
LLAP = **\{**
LLCIERRE = **\}**
FLECHA = **->**
COMA = **\,**
PUNTO = **\.**
ET = **&**

— Palabras reservadas:
R_INT ≡ **int**
R_REAL ≡ **real**
R_BOOL ≡ **bool**
R_TRUE ≡ **true**
R_FALSE ≡ **false**
R_AND ≡ **and**
R_OR ≡ **or**
R_NOT ≡ **not**
R_STRING = **string**
R_NULL = **null**

R_PROC = **proc**
R_IF = **if**
R_THEN = **then**
R_ELSE = **else**
R_ENDIF = **endif**
R_WHILE = **while**
R_DO = **do**
R_ENDWHILE = **endwhile**
R_CALL = **call**
R_RECORD = **record**
R_ARRAY = **array**
R_OF = **of**
R_POINTER = **pointer**
R_NEW = **new**
R_DEL = **del**
R_READ = **read**
R_WRITE = **write**
R_NL = **nl**
R_VAR = **var**
R_TYPE = **type**

II: Clases léxicas multivaluadas

ID ≡ **letra (letra + digito + subrayado)***
letra ≡ **[a-z, A-Z]**
digito ≡ **[0-9]**
subrayado ≡ **_**

LIT_INT ≡ **(+ | -)? parteEntera**
LIT_REAL ≡ **LIT_INT parteDecimal parteExp?**
parteEntera ≡ **digitoPos (digito)* + 0**
parteDecimal ≡ **. (digito* digitoPos + 0)**
digitoPos ≡ **[1-9]**
parteExp ≡ **(e | E) LIT_INT**

III: Cadenas ignorables

SEP = **SB | TAB | NL | CR | BS**
SB ≡ **‘ ‘**
TAB ≡ **\t**
NL ≡ **\n**
BS ≡ **\b**
CR ≡ **\n**

COM = **# [^\n, EOF]***