ANALISTA UNIVERSITARIO EN SISTEMAS

Teoría de Lenguajes

TRABAJO FINAL: Extensión de Lenguaje Imperativo Simple

Alumno: CONO, Diego

1) INTRODUCCIÓN

Se propuso extender dos comandos “Switch-Case” y “Ternario”, y además dos operadores de enteros “linexp” y “promedio”. La propuesta presentada fue la siguiente:

Voy a extender el lenguaje con:
- Los comandos switch-case y ternario de la forma:
comm ::= 'switch' intexp listcase 'end'
listcase ::= 'case' intexp listcomm ';' listcase | 'default' listcomm
listcomm ::= comm ';' listcomm 'end'

comm ::= boolexp ‘?’ comm ‘:’ comm. ‘end’

Donde el comando switch recibe un entero y mediante el comando case lo compara con los distintos casos y si hay coincidencia ejecuta la lista de comandos siguiente, caso contrario ejecuta por defecto los comandos de default.
El segundo comando es un ternario, que funciona como un if_then_else pero en línea

- Y los operadores 'linexp' y 'promedio':

intexp ::= linexp
linexp ::= intexp ',' linexp | var ':=' intexp

intexp ::= 'promedio' listint
listint ::= intexp ',' listint ';'

Donde linexp ejecuta secuencialmente todas las operaciones con enteros separadas con ',' y el ultimo valor la asigna a la variable
Ej:
a = 3
*b = [a*2, a+1, a=5, 7] (b termina con valor 7 y a con 5)*

y el operador 'promedio' suma todos los valores enteros separados por ',' y lo divide por la cantidad total de elementos.

2) SINTAXIS CONCRETA

Comando Switch-Case:

<listcomm> ::= <comm> ‘;’ <listcomm> ‘end’
<listcase> ::= ‘case’ <intexp> <listcomm> ‘;’ <listcase>
| ‘default’ <listcomm>
<comm> ::= ‘switch’ <intexp> <listcase> ‘end’

Comando Ternario:

<comm> ::= <boolexp> ‘?’ <comm> ‘:’ <comm> ‘end’

Operador LinExp:

<linexp> ::= <intexp> ‘,’<linexp>
| <var> ‘:=’ <intexp>
<intexp> ::= ‘[‘ <linexp> ‘]’

Operador Promedio:

<listint> ::= <intexp> ‘,’ <listint> ‘;’
<intexp> ::= ‘promedio’ <listint>

3) SINTAXIS ABSTRACTA

Comando Switch-Case:

<listcomm> ::= <comm> ; <listcomm>
<listcase> ::= case <intexp> <listcomm> ; <listcase>
| default <listcomm>
<comm> ::= switch <intexp> <listcase>

Comando Ternario:

<comm> ::= <boolexp> ? <comm> : <comm>

Operador LinExp:

<linexp> ::= <intexp> ,<linexp>
| <var> := <intexp>
<intexp> ::= [<linexp>]

Operador Promedio:

<listint> ::= <intexp> , <listint>
<intexp> ::= promedio <listint>

4) REGLAS SEMÁNTICAS

Comando Switch-Case:

$$\frac{(e0,\sigma) \Downarrow_{intexp} n0}{\boldsymbol{switch}\ n0\ [\boldsymbol{case}\ (n1, c0): cases]\boldsymbol{default}\ c1 \rightsquigarrow (if\ n0 \equiv n1\ then\ c0\ else\ \boldsymbol{switch}\ n0\ cases\ \boldsymbol{default}\ c1, \sigma)}$$

$$\frac{(e0,\sigma) \Downarrow_{intexp} n0}{\boldsymbol{switch}\ n0\ [\ ]\ \boldsymbol{default}\ c1 \rightsquigarrow (c1,\sigma)}$$

Comando Ternario:

$$\frac{(b,\sigma) \Downarrow_{boolexp} \boldsymbol{true}}{(b\ ?\ c0 : c1, \sigma) \rightsquigarrow (c0,\sigma)} \quad ?1$$

$$\frac{(b,\sigma) \Downarrow_{boolexp} \boldsymbol{false}}{(b\ ?\ c0 : c1, \sigma) \rightsquigarrow (c0,\sigma)} \quad ?2$$

Operador LinExp:

$$\frac{(c0,\sigma) \rightsquigarrow (c'0,\ \sigma')}{(c0: c1, \sigma) \rightsquigarrow (c'0: c1,\ \sigma')} \quad LinExp1$$

$$\frac{(e0,\sigma) \Downarrow_{intexp} n0}{(\boldsymbol{skip};\ n0, \sigma) \rightsquigarrow (n0,\sigma)} \quad LinExp2$$

Operador Promedio:

$$\frac{([es],\sigma) \Downarrow_{intexp} [ns]}{\dfrac{\boldsymbol{plus}[n0: n1: ns] \rightsquigarrow n0 + n1 + (\boldsymbol{plus}[n]) \rightsquigarrow (s,\sigma)}{\boldsymbol{count}[n0: ns] \rightsquigarrow 1 + \boldsymbol{count}[ns] \rightsquigarrow (c,\sigma)}}{\boldsymbol{promedio}[ns] \rightsquigarrow (s \div c, \sigma)}$$

5) EXPLICACIÓN DE LOS COMANDOS

Comando Switch-Case:

El comando switch-case toma una variable que tiene almacenado un valor entero y comienza a compararlo con los valores de cada 'case', cuando encuentra una coincidencia ejecuta la lista de comando del 'case' y termina el comando. En el caso de no encontrar coincidencia ejecuta la lista de comandos del 'default'.

Ejemplo:

```
a:=1;
switch a
case 1: a:=2
case 2: a:=3
default a:=4
end
```

En este caso a terminaría con valor 2.

Comando Ternario:

Este comando recibe una valor booleano y si es 'true' ejecutar el comando que sigue al '?' caso contrario ejecuta el comando seguido a ':'.

Ejemplo:
e:=5;
b:=3;
e<b ? e:=1 : e:=3 end

Para este ejemplo e termina con valor 3 ya que evalúa a 'falso'.

Operador LinExp:

Este operador toma una lista de comandos y un número entero al final. Primero ejecuta todos los comandos y por último asigna a la variable el número entero.

Ejemplo:

b := linexp [c:=3, d:=3, 15]

Aquí, 'c' y 'd' terminan con valor 3 y con 'b' con valor 15.

Operador Promedio:

Este operador toma una lista de números enteros y devuelve su promedio, o sea, suma todos sus valores y los divide por la cantidad de elementos.

e := promedio 3 8 10 5 9;

La variable 'e' finaliza con valor 7.