

# **APUNTES DE LOGO**

Eugenio Roanes Lozano<sup>1</sup> y Eugenio Roanes Macías<sup>2</sup>

Unidad Docente de Álgebra<sup>3</sup>  
Facultad de Educación  
Universidad Complutense de Madrid

26 Diciembre 2015

---

<sup>1</sup> [eroanes@mat.ucm.es](mailto:eroanes@mat.ucm.es)

<sup>2</sup> [roanes@mat.ucm.es](mailto:roanes@mat.ucm.es)

<sup>3</sup> [www.ucm.es/info/secdealg/](http://www.ucm.es/info/secdealg/)

## **DESCARGA DE LOGO:**

Una breve resumen de la historia de Logo (y sus dialectos) y la posibilidad de descargar **MSWLogo 6.3** (funciona hasta Windows XP, no compatible con Windows 7 y posteriores):

<http://roble.pntic.mec.es/~apantoja/familias.htm>

Descarga de **MSWLogo 6.5a** (compatible con Windows modernos) (**el más recomendable**<sup>4</sup>)

<http://neoparaiso.com/logo/versiones-logo.html>

Descarga de **FMSLogo 6.34.0** (compatible con Windows modernos)<sup>5</sup>:

<http://neoparaiso.com/logo/versiones-logo.html>

## **ALGUNAS OBSERVACIONES**

Ambos dialectos derivan de UCBLogo, diseñado por Brian Harvey, de la Universidad de Berkeley. Los tres volúmenes de su libro “Computer Science Logo Style”, publicado en 1997 por el MIT e información sobre Logo puede encontrarse en:

<http://www.cs.berkeley.edu/~bh/>

El libro pionero y enciclopédico sobre las posibilidades de la geometría de la tortuga es:

- Abelson, H. & di Sessa, A. (1986). Geometría de tortuga: el ordenador como medio de exploración de las Matemáticas. Madrid, España: Anaya.

Un libro de esa época en que se desarrollan diferentes aplicaciones educativas para antiguos dialectos de Logo es:

- Roanes Macías, E. & Roanes Lozano, E. (1988). MACO. Matemáticas con ordenador. Madrid, España: Síntesis.

Es de destacar que hay localizados más de 300 dialectos de Logo. Véase:

<http://www.elica.net/download/papers/LogoTreeProject.pdf>

## **INSTALACIÓN:**

Sólo hay que ejecutar el correspondiente archivo .EXE.

## **ARRANQUE DE MSWLogo6.5a:**

- 1) Hacer click en el icono correspondiente.
- 2) Hacer click en OK
- 3) Elegir “programar” y hacer click en OK.  
(obsérvese que la línea de input es la de más abajo).

---

<sup>4</sup> MSWLogo sí tiene el inconveniente de que si se elige un tamaño grande de letra para las órdenes, no aumenta adecuadamente la altura de la línea de input.

<sup>5</sup> FMSLogo en español tiene el problema de que funciona tanto con primitivas en inglés como en español, lo que provoca interferencias y además faltan abreviaturas de comandos. Por ejemplo

- PU no es abreviación de PONULTIMO, sino de PENUP. Hay que usar PONULTIMO o LPUT.
- MT no es abreviación de MUESTRATORTUGA (no funciona). Hay que usar MUESTRATORTUGA, SHOWTURTLE o ST (abreviatura ésta última de SHOWTURTLE).

## Hoja 1: INICIACION A LA GEOMETRIA DE LA TORTUGA

<u>Descripción de la primitiva o comando</u>	<u>castellano</u>	<u>inglés</u>
AVanzar n pasos (pixels aprox.)	<b>AV n</b>	<b>FD n</b>
REtroceder n pasos	<b>RE n</b>	<b>BK n</b>
Girar Derecha n grados (sentido horario)	<b>GD n</b>	<b>RT n</b>
Girar Izquierda n grados (sentido antihorario)	<b>GI n</b>	<b>LT n</b>

Borrar Pantalla (y tortuga al centro)	<b>BP</b>	<b>CS</b>
---------------------------------------	-----------	-----------

No todas las primitivas son necesarias, p.ej.: **GI 90** equivale a: **GD -90**  
y: **RE 100** equivale a: **AV -100**

Subir Lápiz (no dejar traza al moverse)	<b>SL</b>	<b>PU</b>
Bajar Lápiz (dejar traza al moverse)	<b>BL</b>	<b>PD</b>
Ocultar la Tortuga	<b>OT</b>	<b>HT</b>
Mostrar la Tortuga	<b>MT</b>	<b>ST</b>
Cambiar a color n (n=1,2,...,16) (n=0..255 en MSW-Logo)	<b>PONCL n</b>	<b>SETPC n</b>
Borrar por donde va pasando	<b>GOMA</b>	<b>PE</b>
Dejar de borrar por donde va pasando (en MSW-Logo: o bien:	<b>BL</b> <b>LAPIZNORMAL</b> <b>PONLÁPIZ</b>	<b>PD</b>

Repetir n veces lo indicado entre corchetes:	<b>REPITE n [...] REPEAT n [...]</b>
--	--------------------------------------

Para rellenar el interior de una línea simple cerrada,  
previamente trazada, se lleva la tortuga a un punto  
interior y se ordena:

**RELLENA**                      **FILL**

Nota: en la mayoría de los Logos el relleno se realiza en el color  
del lápiz y puede “escaparse” si el borde no es del mismo color;  
en MSW-Logo el color de relleno se cambia con:

**PONCOLORRELLENO n**

Si la tortuga se sale de los límites de pantalla, en MSWLogo  
“reaparece por el lado opuesto”. Para pasar al modo en que vemos  
una porción del plano que contiene a la pantalla:

**MODOVENTANA**                      **WINDOW**

Volver al modo “por defecto”:

**MODOVUELTA**                      **WRAP**

<sup>6</sup> En algunos Logos españoles, las primitivas se tienen que acentuar con las reglas usuales del castellano. En MSWLogo funcionan con o sin tilde.

## Hoja 1: INICIACION A LA GEOMETRIA DE LA TORTUGA (Continuación)

---

### EJERCICIOS:

1. Dibujar el borde de un cuadrado en sentido antihorario.
  2. Dibujar el borde de un rectángulo en sentido horario.
  3. Dibujar el borde de un triángulo equilátero en sentido antihorario.
  4. Dibujar el borde de un cuadrado de centro el centro de pantalla.
  5. Dibujar el borde de un hexágono regular.
  6. Dibujar el borde de un cuadrado, haciendo uso de Repite.
  7. Dibujar el borde de un triángulo equilátero, haciendo uso de Repite.
  8. Dibujar el borde de un hexágono regular, haciendo uso de Repite.
  9. Dibujar un rectángulo haciendo uso de repite.
  10. Dibujar un rectángulo rellenando su interior.
  11. Dibujar los radios de una rueda con 24 radios.
  12. \*<sup>7</sup> Dibujar el borde de un hexágono regular y las diagonales de extremos cada dos vértices opuestos del hexágono (hay, al menos, tres formas distintas de hacerlo).
- 

---

<sup>7</sup> Los ejercicios marcados con un asterisco tienen cierta dificultad. Los marcados con dos asteriscos son más complejos.

## Hoja 2: PROCEDIMIENTOS Y SUBPROCEDIMIENTOS

---

Procedimiento titulado CUADRADO, que dibuje un cuadrado de lado fijo:

<b>PARA CUADRADO</b>	<b>TO CUADRADO</b>
<b>REPITE 4 [AV 77 GD 90]</b>	<b>REPEAT 4 [FD 77 RT 90]</b>
<b>FIN</b>	<b>END</b>

---

Procedimiento titulado RECTANG, que dibuje un rectángulo de lados fijos:

<b>PARA RECTANG</b>
<b>REPITE 2 [AV 50 GD 90 AV 100 GD 90]</b>
<b>FIN</b>

---

Programa que dibuje una colección de cuadrados con un vértice común, cada uno de ellos girado 36 grados respecto del anterior (llamaremos COLECUAD al procedimiento y CUADRADO al subprocedimiento):

<b>PARA COLECUAD</b>
<b>REPITE 10 [CUADRADO GI 36]</b>
<b>FIN</b>

<b>PARA CUADRADO</b>
<b>REPITE 4 [AV 35 GD 90]</b>
<b>FIN</b>

---

Programa que dibuje una fila horizontal de cinco cuadrados:

<b>PARA FILACUAD</b>
<b>REPITE 5 [CUADRADO SL GD 90 AV 45 GI 90 BL]</b>
<b>FIN</b>

---

Si se desea incluir comentarios en un procedimiento, basta preceder la(s) línea(s) de ;

---

### EJERCICIOS

1. Procedimiento TRIANG, que dibuje el borde de un triángulo equilátero.
  2. Procedimiento HEXAG, que dibuje el borde de un hexágono regular.
  3. Procedimiento que dibuje un cuadrado de centro el centro de pantalla.
  4. Procedimiento que dibuje los 24 radios de una rueda de bicicleta (sólo los radios).
  5. Programa que dibuje una colección de hexágonos regulares con un vértice común, cada uno de ellos girado 60 grados respecto del anterior.
  6. \* Programa que dibuje una fila horizontal de cinco triángulos equiláteros, cuyas bases están contenidas en la misma recta.
  7. \*. En el procedimiento anterior, si consideramos un triángulo y los adyacentes a él, pueden compartir un vértice, cortarse o están separados. Pensar cuando se da cada caso.
-

### Hoja 3: VARIABLES. PROCEDIMIENTOS CON ENTRADAS

---

Para guardar en la variable AB el número 32.7:

**HAZ "AB 32.7**

**MAKE "AB 32.7**

Para guardar en la variable G la palabra GATO:

**HAZ "G "GATO**

**MAKE "G "GATO**

Para guardar en la variable F la FRASE *el gato gris*:

**HAZ "F [EL GATO GRIS]**

**MAKE "F [EL GATO GRIS]**

---

Para pedir lo almacenado en la variable V:     **:V**

y para escribirlo<sup>8</sup>:     **ES :V**

**PR :V**

Si lo almacenado es un número, se puede hacer con ello lo mismo que con un número:

**AV :AB**

**FD :AB**

**GD :AB**

**RT :AB**

---

Un procedimiento poco elegante<sup>9</sup> que dibuja un cuadrado cuyo lado mide el número almacenado en la variable L es:

**PARA CUADRADO**

**REPITE 4 [AV :L GD 90]**

**FIN**

Por ejemplo, para dibujar dos cuadrados, de lados 77 y 82, se puede proceder así:

**HAZ "L 77**

**CUADRADO**

**HAZ "L 82**

**CUADRADO**

Pero es más breve y elegante<sup>10</sup>, construirlo como sigue (procedimiento con una entrada):

**PARA CUADRADO :L**

**REPITE 4 [AV :L GD 90]**

**FIN**

en cuyo caso se ejecuta de este modo:

**CUADRADO 77**

**CUADRADO 82**

---

<sup>8</sup> Detallaremos más adelante cómo escribir números, palabras y frases.

<sup>9</sup> Haciéndolo así la variable sería *global* (habría que reservarla para uso exclusivo de este procedimiento).

<sup>10</sup> Varios procedimientos pueden tener una entrada con el mismo nombre (son variables *locales*). Además, estas entradas no interferirían con una variable externa a los procedimientos que tuviera el mismo nombre.

### Hoja 3: VARIABLES. PROCEDIMIENTOS CON ENTRADAS (Continuación)

---

Un procedimiento con entradas "transfiere" los valores de las variables a un subprocedimiento suyo.

```
PARA CUADRADOS :L  
REPITE 8 [CUADRADO :L GD 45]  
FIN
```

Ejecución: **CUADRADOS 50**

---

Procedimiento que dibuje un rectángulo cuyos lados midan los números almacenados en las variables BAS y ALT:

```
PARA RECTANG :BAS :ALT  
REPITE 2 [AV :ALT GD 90 AV :BAS GD 90]  
FIN
```

Ejecución de este procedimiento: **RECTANG 100 60**

---

#### EJERCICIOS

1. Procedimiento TRIANG, que dibuje un triángulo equilátero de lado L.
  2. Procedimiento RUEDA, que dibuje los 36 radios de longitud L de una rueda.
  3. Procedimiento BANDERA, que dibuje una bandera consistente en un mástil de longitud L y cuya tela sea rectangular de lados A y B.
  4. Programa que dibuje una fila horizontal de M baldosas cuadradas (no separadas ni solapadas) de lado L
  5. \* Programa que dibuje una fila de N ruedas (de 36 radios de longitud L), de modo que la distancia entre los centros de dos contiguas sea D.
-

## Hoja 4: GEOMETRIA DE LA TORTUGA 3D

---

Nota: Las primitivas de tortuga 3D sólo existen en algunos Logos. Las de esta hoja corresponden al dialecto MSW-Logo<sup>11</sup> y se describen considerando a la tortuga como si fuera un avión.

Pasar al modo 3D:	<b>PERSPECTIVA</b>
Volver al modo 2D:	<b>MODOVENTANA</b>

Nota: Al pasar al modo 3D aparece una raya en el ala derecha del avión (para distinguir su posición cuando está, por ejemplo, en vuelo invertido).

Girar a derecha (resp. izquierda) n grados en el plano que contiene al triangulito que representa a la tortuga:	<b>GD</b>
(resp.:	<b>GI )</b>

Sube el morro n grados:	<b>CABECEA n</b>
Baja el morro n grados:	<b>BAJANARIZ n</b>
Sube ala izquierda y baja ala derecha n grados:	<b>BALANCEA n</b>
(En FMSLogo:	<b>BALANCEADERECHA n )</b>
Sube ala derecha y baja ala izquierda n grados:	<b>BALANCEAIZQUIERDA n</b>

---

Borrar Pantalla (para llevar la tortuga al centro, orientada “hacia arriba”, como el transbordador espacial en el despegue, con el timón de cola casi hacia nosotros)	<b>BP</b>
---	-----------

---

Como ocurre en 2D, no todas las primitivas son necesarias, p.ej.:	<b>BAJANARIZ 90</b>
es equivalente a:	<b>BALANCEA 90 GD 90 BALANCEAIZQUIERDA 90</b>

---

A partir de la posición inicial, tras ejecutar **BP**, podemos dibujar los semiejes en la posición habitual tecleando, por ejemplo:

semieje z+:	<b>AV 200 RE 200</b>
semieje x+:	<b>BAJANARIZ 90 RE 200 AV 200</b>
semieje y+:	<b>GD 90 AV 200 RE 200</b>

---

Las siguientes líneas de comandos dibujan cuadrados en distintos planos verticales:

<b>BP REPITE 4 [AV 100 GD 90]</b>
<b>BP REPITE 4 [AV 100 CABECEA 90]</b>

---

Podemos dibujar 8 cuadrados verticales, cada uno girado respecto del anterior 45 grados, así:

<b>BP REPITE 8 [BALANCEA 45 REPITE 4[AV 100 GD 90]]</b>
---

---

Pedir las lista de las coordenadas de la tortuga en el espacio:	<b>POS3D</b>
---	--------------

---

---

<sup>11</sup> La implementación de primitivas 3D en WinLogo es completamente distinta. Por ejemplo, no existe la primitiva **PERSPECTIVA** (las primitivas 3D funcionan directamente); los nombres de algunas de las primitivas 3D son distintos; y **AV** ha sido sustituido en 3D por **ANDA**.



## Hoja 4: GEOMETRIA DE LA TORTUGA 3D (Continuación)

---

### EJERCICIOS:

1. Dibujar un taburete cuadrado con una pata en cada esquina. Idea: repetir 4 veces “lado – pata”.
  2. \* A partir del taburete anterior, dibujar una silla (con el respaldo inclinado).
  3. \* Procedimiento con una entrada que, reformando el anterior, permita variar la inclinación del respaldo respecto de la vertical.
  4. \* Reformar el procedimiento anterior de forma que se pueda cambiar también la altura del respaldo.
  5. \* Reformar la silla anterior para dibujar una tumbona.
  6. \* Dibujar un cubo.
  7. \*\* Dibujar una pirámide cuadrangular regular. Idea: proceder como si fuera un taburete invertido, con las patas inclinadas 45 grados respecto del plano horizontal.
  8. \*\* Dibujar un octaedro regular a partir de la pirámide anterior.
-

## Hoja 5: OPERACIONES

---

### Operaciones binarias

Para sumar 13, 7 y 2.5:	<b>13 + 7 + 2.5</b>	
y para escribir su resultado:	<b>ES 13 + 7 + 2.5</b>	<b>PR 13 + 7 + 2.5</b>
o para avanzar esa distancia:	<b>AV 13 + 7 + 2.5</b>	<b>FD 13 + 7 + 2.5</b>
Para multiplicar 4 por 2.5:	<b>4 * 2.5</b>	
Para restar 13 menos 7:	<b>13 - 7</b>	
Para dividir 15.7 entre 10:	<b>15.7/10</b>	
Cociente entero de 17 entre 5:	<b>COCIENTE 17 5</b>	<b>INTQUOTIENT 17 5</b>
Resto de división entera 17 entre 5:	<b>RESTO 17 5</b>	<b>REMAINDER 17 5</b>

Observación importante: **COCIENTE** funciona incorrectamente en FMSLogo (devuelve el mismo resultado que **/**) y NO EXISTE en MSWLogo.

---

Como es habitual, los paréntesis indican la prioridad entre operaciones: **3\*(2+4)**

Si no se ponen paréntesis, se sigue el orden de prioridades interno de Logo. Por ejemplo: **3\*2+4** se interpreta como **(3\*2)+4**, puesto que, para Logo, el producto es prioritario frente a la suma.

---

Procedimiento que calcula el área de un triángulo de base B y altura H:

```
PARA AREA :B :H
HAZ "AR (:B * :H) / 2
ES :AR
FIN
```

Para ejecutarlo se procede así: **AREA 5 3**

Pero es más elegante incluir antes de la penúltima línea: **ES [SU AREA ES]**

o bien sustituirla por: **(ES [SU AREA ES] :AR)**

(esto último para que aparezcan ambas cosas en la misma línea de output).

---

### Operaciones unarias:

Para calcular la raíz cuadrada de 64:	<b>RC 64</b>	<b>SQRT 64</b>
Para calcular la parte entera de 5.13:	<b>ENTERO 5.13</b>	<b>INT 5.13</b>
Para redondear el número 5.13:	<b>REDONDEA 5.13</b>	<b>ROUND 5.13</b>
Para número natural aleatorio $\geq 0$ y $< 15$ :	<b>AZAR 15</b>	<b>RANDOM 15</b>

---

Procedimiento que calcula el cociente y el resto de la división entera del número A entre el B (sin usar las primitivas **COCIENTE** y **RESTO**):

```
PARA DIVENT :A :B
HAZ "C ENTERO (:A / :B)
ES :C
ES :A - :B * :C
FIN
```

---

Procedimientos que simulan la tirada de un dado y de una moneda con un 5 en una cara y un 0 en la otra (respectivamente):

<b>PARA DADO</b>	<b>PARA MONEDA</b>
<b>ES 1 + AZAR 6</b>	<b>ES 5 * AZAR 2</b>
<b>FIN</b>	<b>FIN</b>

---

## Hoja 5: OPERACIONES (Continuación)

---

### Otras operaciones binarias:

Para calcular 3 elevado a 5 (sólo algunos Logos, por ejemplo MSW-Logo):

**POTENCIA 3 5**

### Otras operaciones unarias:

Funciones trigonométricas directas:  
(ángulo expresado en grados)

**SEN 45**

**SIN 45**

**COS 45**

**COS 45**

**TAN 45**

**TAN 45**

e inversa:

**ARCTAN 1**

**ARCTAN 1**

(en MSW-Logo y WinLogo existen también **ARCSEN** y **ARCCOS**).

---

## EJERCICIOS

- 1 Calcular  $5 - 3 - 2$  para deducir cómo Logo pone los paréntesis que precisa esa operación compuesta.
- 2 Reformar el procedimiento DIVENT de modo que devuelva los resultados indicando con mensajes cuál es el cociente de la división entera y el resto.
- 3 Procedimiento PITAG, que calcule la medida de la hipotenusa de un triángulo rectángulo cuyos catetos midan X e Y.
- 4 Procedimiento RAYOS, que dibuje los N radios de longitud L de una rueda (sólo los radios).
- 5 Procedimiento POLREG, que dibuje un polígono regular de N lados de longitud L (cada lado).
- 6 \* Procedimiento SUERTE, que saque al azar uno de los números siguientes<sup>12</sup>: 20, 25, 30, 35, 40, 45, 50.
- 7 \* Procedimiento DADOS que simule la suma de la tirada de dos dados.
- 8 \*\* Procedimiento POTEN :X :N que calcule  $x^n$  (supuesto que n es un número natural mayor que 1), sin usar la primitiva **POTENCIA**.  
Nota: Usar **REPITE** y una variable para ir guardando los resultados parciales.
- 9 \*\* Hacer algo análogo para obtener el producto por un número natural no nulo a partir de la operación suma.
- 10 \*\* Hacer algo análogo para obtener la operación suma a partir de la operación elemental “sumar 1”.  
Nota: Los ejercicios 7,8,9 corresponden con la idea de la axiomática de Peano.
- 11 \*\* Procedimiento FACTORIAL :N que calcule la factorial de n.  
Nota: Usar **REPITE**, una variable "contador" y otra variable para ir guardando los resultados parciales.
- 12 \*\* Procedimiento SUMPOT :N que calcule  $2+4+8+\dots+2^n$  usando la primitiva **POTENCIA**.

Nota: Implementaremos de otro modo más elegante los ejercicios 7, 8 y 9 en la hoja 7.

---

<sup>12</sup> Obsérvese que **AZAR 5 + 1000** es lo mismo que: **AZAR (5 + 1000)** y que: **AZAR 1005**, pues la suma es prioritaria frente a **AZAR**. Son equivalentes: **1000 + AZAR 5** y: **(AZAR 5) + 1000**

## Hoja 6: COORDENADAS Y RUMBO

---

Para pedir la posición de la tortuga (abscisa, ordenada), siendo el origen de coordenadas el centro de pantalla gráfica y la unidad de longitud un “paso” de la tortuga:

	<b>POS</b>	<b>POS</b>
Para pedir sólo su abscisa:	<b>COORX</b>	<b>XCOR</b>
Y para pedir sólo su ordenada:	<b>COORY</b>	<b>YCOR</b>

---

Para que la tortuga se traslade (en línea recta) al punto (100,60):

**PONPOS [100 60]**      **SETPOS [100 60]**

Para que se traslade al punto de abscisa 30 (sin cambiar la ordenada):

**PONX 30**      **SETX 30**

Para que se traslade al punto de ordenada 25 (sin cambiar la abscisa):

**PONY 25**      **SETY 25**

(estas primitivas no alteran la orientación de la tortuga).

---

Nota: Como se habrá observado, las coordenadas del punto (30,20) son dadas a Logo en la forma [30 20]. Pero si el valor de la abscisa está guardado en la variable A y el de la ordenada en la variable B, entonces se han de dar así:      **FR :A :B**      **SE :A :B**

(lo que se justificará más adelante, al manejar listas).

---

Para pedir el rumbo de la tortuga:

**RUMBO**      **HEADING**

El rumbo se mide en grados, respecto de la vertical hacia arriba y en sentido horario, es decir, respecto del semieje y+ (como lo haría el timonel de un barco colocado como lo está la tortuga):

Para orientar la tortuga con rumbo 45 grados:

**PONRUMBO 45**      **SETH 45**

Para orientar la tortuga hacia el punto de coordenadas (60,40):

**PONRUMBO HACIA [60 40]**      **SETH TOWARDS [60 40]**

(estas primitivas no alteran la posición de la tortuga -sus coordenadas-, sólo su orientación).

Nota: De hecho, **HACIA** lo que hace es calcular el ángulo hacia el punto de destino a partir de las coordenadas del punto en que está la tortuga y las del punto destino (probablemente usando **ARCTAN**), para pasarle el valor calculado a **PONRUMBO**. Se puede comprobar, por ejemplo, haciendo: **BP HACIA [100 100]**, que devuelve: **45**.

---

## Hoja 6: COORDENADAS Y RUMBO (Continuación)

---

### EJERCICIOS

- 1 Dibujar un rectángulo usando sólo las primitivas **PONX** y **PONY**.
  - 2 Dibujar un triángulo rectángulo isósceles usando sólo **PONPOS**.
  - 3 Dibujar un triángulo rectángulo de catetos de longitud 100 y tal que su hipotenusa está contenida en el eje x.
  - 4 Construir el procedimiento **SEGMENTO**, cuyas 4 entradas sean las coordenadas de 2 puntos y que dibuje el segmento de extremos dichos puntos.
  - 5 Definir el procedimiento **CUADRILA**, cuyas 8 entradas sean las coordenadas de 4 puntos y dibuje el cuadrilátero de vértices dichos puntos.
  - 6 Definir el procedimiento **DISTAN**, cuyas 4 entradas sean las coordenadas de 2 puntos, que calcule y devuelva la distancia entre ellos<sup>13</sup>.
  - 7 Definir unos procedimientos **N**, **S**, **E**, **W**, **NE**, **NW**, **SE**, **SW**, que pongan a la tortuga rumbo a estos puntos cardinales.
  - 8 \*\* Definir el procedimiento **TRIAN**, cuyas 6 entradas sean las coordenadas de 3 puntos y que dibuje el triángulo de vértices dichos puntos y lo rellene.
  - 9 \*\* Definir un procedimiento **ANG :A1 :A2 :B1 :B2** que mida, usando comandos de esta Hoja 6, el ángulo que forman la recta que pasa por (0,0) y (A1,A2) con la que pasa por (0,0) y (B1,B2). Se supone que los dos puntos están en el primer cuadrante.
  - 10 Comparar el efecto de **GD/GI** con **PONRUMBO** y **PONRUMBO HACIA** sobre varias tortugas situadas en distintos puntos de la pantalla.
- 

---

<sup>13</sup> Obsérvese que con este procedimiento y las primitivas **PONRUMBO HACIA** y **AV** no haría falta la primitiva **PONPOS**

## Hoja 7: CONDICIONALES Y OPERACIONES LÓGICAS

---

Vimos cómo escribir un número:	<b>ES 7</b>	<b>PR 7</b>
pero también se pueden escribir palabras:	<b>ES "HOLA</b>	<b>PR "HELLO</b>
y frases:	<b>ES [Hola Pepe]</b>	<b>PR [Hi Joe]</b>

---

Procedimiento que compara un número dado con 4 y, si el número es mayor que 4, contesta *MAYOR*:

```
PARA MAY :X  
SI :X > 4 [ES "MAYOR]  
FIN
```

---

Procedimiento que compara un número con 4, para ver si es mayor que 4 o no lo es:

```
PARA COMPARA :X  
SISINO :X > 4 [ES "SI] [ES "NO]  
FIN
```

Nota: En MSW-Logo el condicional, cuando va seguido de dos corchetes, es **SISINO**, en lugar de **SI** (si va seguido de un sólo corchete, es el habitual **SI**). En otros dialectos de Logo es siempre **SI**

---

Procedimiento que decide si un número es cero, o no lo es:

```
PARA DECIDE :N  
SISINO :N = 0 [ES "SI] [ES "NO]  
FIN
```

---

Procedimiento que decide si la edad no es suficiente para poder entrar:

```
PARA EDAD :E  
SI NO :E < 18 [ES [PUEDES ENTRAR]]  
FIN
```

---

Procedimiento que, si se verifican las dos condiciones  $T > 18$  y  $T < 25$ , indica que la temperatura es agradable:

```
PARA TEMP :T  
SI Y :T > 18 :T < 25 [ES "AGRADABLE]  
FIN
```

---

Procedimiento que, si se verifica una (al menos) de las dos condiciones  $E > 70$  o  $E < 6$ , informa que la edad permite asistir gratis a cierto espectáculo:

```
PARA NOPAGAR :E  
SI O :E > 70 :E < 6 [ES "GRATIS]  
FIN
```

---

Los equivalentes en los Logos ingleses de **SI**, **NO**, **Y**, **O**, son, respectivamente: **IF**, **NOT**, **AND**, **OR**

---

## Hoja 7: CONDICIONALES Y OPERACIONES LÓGICAS (Continuación)

---

Para parar la ejecución de un procedimiento (saltar directamente al **FIN** o **END**), se puede incluir en él la orden:

**ALTO**

**STOP**

pero es mejor evitar su uso (es poco elegante). Un sencillo ejemplo de procedimiento que lo usa es:

```
PARA CALOR :T
SI :T>29 [ES [QUE CALOR] ALTO]
SI :T>25 [ES [NO HACE FRIO]]
FIN
```

---

Para que al ejecutar un procedimiento, éste devuelva el valor alojado en la variable X, basta incluir en el la orden: **DEV :X**. Así el procedimiento se comporta exactamente igual que una operación.

El procedimiento siguiente devuelve el resultado de añadir al precio (P) el porcentaje de IVA (I) que se especifique:

```
PARA CONIVA :P :I
DEV :P * (1 + :I / 100)
FIN
```

y permite hacer lo siguiente (lo que no sería posible usando **ES** en lugar de **DEV**):

```
ES (CONIVA 134 7 + CONIVA 13 16)
```

---

Un procedimiento que se llama a sí mismo (esto es, que es subprocedimiento de si mismo) se denomina *procedimiento recursivo*. Una definición *recursiva* típica en Matemáticas es la de factorial. En lugar de definir:  $n! = n \cdot (n-1) \cdot \dots \cdot 2 \cdot 1$ , se puede definir:

$$n! = \begin{cases} 1 & , \quad \text{si } n=0 \\ n \cdot ((n-1)!) & , \quad \text{si } n>0 \end{cases}$$

En MSW-Logo:

```
PARA FACT :N
SISINO :N=0 [DEV 1] [DEV :N * FACT(:N - 1)]
FIN
```

---

### EJERCICIOS

1. Procedimiento NOMENOR, que decida si, dados dos números, el primero es mayor o igual al segundo, respondiendo “sí” en caso afirmativo.
  2. Procedimiento EDAD, que compruebe si la edad, e, de una persona verifica la condición  $17 < e < 65$ , respondiendo “sí” en caso afirmativo.
  3. Procedimiento EDAD, que compruebe si la edad, e, de una persona verifica la condición  $17 \leq e < 65$ , respondiendo “sí” en caso afirmativo.
  4. Procedimiento MULTIPLO, que verifique si el número A es múltiplo del B (usando negación), respondiendo “sí” en caso afirmativo y “no” en caso negativo.  
Nota: Se puede hacer usando **COCIENTE** (no en MSWLogo, ni FMSLogo) o bien usando **RESTO** o bien usando la operación división (/).
  5. Procedimiento SIG, que decida si dos números no nulos son del mismo signo (comparando su producto con 0).
  6. Procedimiento DESCUENTO con dos entradas que aplique a la primera entrada (precio) el % de descuento indicado por la segunda entrada.
-

## Hoja 7: CONDICIONALES Y OPERACIONES LÓGICAS (Continuación)

---

7. \* Comprobar que añadir IVA y hacer un descuento permutan. Pensar la razón.
  8. Procedimiento MENIG que decida si un número es menor que 100, igual a 100 o mayor que 100, no haciendo preguntas innecesarias en ningún caso (sugerencia: usar ALTO).
  9. \* Procedimiento CALIFICA, que califique la nota  $x$  de acuerdo con el baremo usual: suspenso si  $x < 5$ , aprobado si  $5 \leq x < 6.5$ , notable si  $6.5 \leq x < 8.5$ , sobresaliente si  $8.5 \leq x < 9.5$  y matrícula de honor si  $x \geq 9.5$ .  
Nota: se puede hacer con las preguntas consecutivas simplemente, con las preguntas consecutivas y usando ALTO, o con las preguntas anidadas.
  10. Programa que calcule la hipotenusa de un triángulo rectángulo, dados sus dos catetos, usando un subprocedimiento que devuelva el cuadrado de un número dado.
  11. \*\* Programa recursivo que calcule potencias de exponente natural positivo.
  12. \*\* Programa recursivo que calcule potencias de exponente natural.
  13. \*\* Programa recursivo que calcule el producto de dos números naturales a partir de la operación  $+$ .
  14. \*\* Programa recursivo que calcule la suma de dos números naturales a partir de la operación “sumar 1”.
  15. \*\* Programa recursivo que calcule el término  $n$ -ésimo de la sucesión de Fibonacci. Esta sucesión es la: 1,1,2,3,5,8,13,21,34,55,... (comienza: 1,1,... y cada término se obtiene sumando los dos anteriores).
  16. \*\* Programa recursivo que devuelva la suma de los  $n$  primeros pares (excluido el cero), esto es:  $2+4+6+8+\dots+2 \cdot n$  (Nota: aplicar la fórmula de la suma de términos de una progresión aritmética no es lo que se pide).
  17. \*\* Programa recursivo que devuelva la suma de los  $n$  primeros términos de la progresión aritmética de primer término 5 y razón 3, esto es: 5, 8, 11, 14, 17,... (Nota: aplicar la fórmula de la suma de términos de una progresión aritmética no es lo que se pide).
  18. \*\* Programa CUADRADOS, con una entrada,  $L$ , que, cuando se ordene CUADRADOS 15, dibuje cuadrados de lados 15, 25, 35, 45, 55, 65 y 75 (estos cuadrados deben tener todos un vértice común y sus lados paralelos). Este programa debe ser recursivo y debe ir enviando a un subprocedimiento CUADRADO : $L$ , cada vez incrementando  $L$  en 10 en 10. Debe usar un condicional para parar (al sobrepasarse el valor 75).
  19. \*\* Programa CUA1000, que tenga una entrada,  $N$ , y escriba los números naturales que son menores que 1000 y cuadrados de un natural. (Idea: no ir comprobando qué números son cuadrados de natural, sino generar con un programa recursivo los cuadrados de los sucesivos naturales y que no pare mientras estos cuadrados se mantengan menores que 1000).
-



## Hoja 8: LISTAS

[53 gato [7 28] 4.9] es una lista en Logo.

Su primer elemento es 53, el segundo gato, el tercero [7 28] y el cuarto 4.9.

Se aloja en la variable L así:

**HAZ "L [53 gato [7 28] 4.9]                      MAKE "L [53 gato [7 28] 4.9]**

Primer elemento de la lista L:	<b>PRI :L</b>	<b>FIRST :L</b>
Último elemento de la lista L:	<b>UL :L</b>	<b>LAST :L</b>
Elemento n-ésimo de la lista L:	<b>ELEMENTO n :L</b>	<b>ITEM n :L</b>
Número de elementos de la lista L:	<b>CUENTA :L</b>	<b>COUNT :L</b>

Eliminar el primer elemento de L:	<b>MP :L</b>	<b>BF :L</b>
Eliminar el último elemento de L:	<b>MU :L</b>	<b>BL :L</b>
Eliminar el elemento x de L	<b>ELIMINA x :L</b>	(elimina su primera ocurrencia)

(este comando no es habitual; p. ej. existe en WinLogo, pero no en MSWLogo ni en FMSLogo)

Anteponer x como primer elemento de la lista L:	<b>PP x :L</b>	<b>FPUT x :L</b>
Anteponer x como último elemento de la lista L:	<b>PU x :L</b>	<b>LPUT x :L</b>
Intercalar x como elemento n-ésimo de L:	<b>INSERTA n x :L</b>	

(este comando no es habitual; p. ej. existe en WinLogo, pero no en MSWLogo ni en FMSLogo)

Concatenar (enristrar) las listas L y M, ya definidas:

**FR :L :M                      SE :L :M**

Nota: Si la abscisa de un punto está almacenada en la variable P1 y su ordenada en la variable P2, entonces, para enviar la tortuga al punto de tales coordenadas, se ordena (de acuerdo con lo indicado anteriormente):

**PONPOS FR :P1 :P2      SETPOS SE :P1 :P2**

De modo similar, podemos escribir:

**PONRUMBO HACIA FR :P1 :P2      SETH TOWARDS SE :P1 :P2**

Si se desea que, al ejecutar un procedimiento, éste se detenga para “leer una lista”, alojándola en la variable H, basta que, al definirlo, se inserte en el lugar apropiado un LEELISTA (LL):

**HAZ "H LL                      MAKE "H RL**

Ejemplo:

```
PARA SEGMENT
ES [ESCRIBE LAS COORDENADAS DEL PUNTO A]
HAZ "A LL
ES [ESCRIBE LAS COORDENADAS DEL PUNTO B]
HAZ "B LL
SL PONPOS :A BL PONPOS :B
FIN
```

se ejecuta tecleando: **SEGMENT** (y pulsando Intro) al pedir el procedimiento las coordenadas de A contestamos, p.ej., **40 50** y pulsamos Intro y al pedir las de B contestamos, p.ej., **100 0** (e Intro).

Observación: Si se recoge en la variable A un número con **LEELISTA**, por ejemplo 100, no se puede hacer:

**AV :A**

pues equivaldría a pedir:

**AV [100]**

sino que hay que ordenar:

**AV PRI :A**

## Hoja 8: LISTAS (Continuación)

---

### EJERCICIOS

1. ¿Cómo se puede extraer el 22 que aparece en la lista de listas **[[22 3] [4 5] [8 35]]**?
  2. Procedimiento **PRIME**, con una entrada, **L**, que devuelva el primer elemento de **L**, sin usar la primitiva **PRI**.
  3. Procedimiento **ULTI**, con una entrada, **L**, que devuelva el último elemento de **L**, sin usar la primitiva **UL**.
  4. Procedimiento **TRI**, que, al ser ejecutado, vaya pidiendo las coordenadas de los vértices de un triángulo (uno a uno) y lo dibuje.
  5. Procedimiento que pida la medida del lado de un cuadrado y devuelva la medida de su diagonal.
  6. \* Procedimiento **MENGUA** que pida escribir una lista y escriba las listas que se obtienen al ir eliminando un elemento cada vez (por ejemplo el último), hasta quedar vacía.
  7. \*\* Procedimiento **INV**, que pida una lista y la devuelva invertida (su orden).
  8. \*\* Programa **MAXIMO**, que pida escribir una lista de números y devuelva el mayor de ellos.
  9. \*\* Programa **SUPRIME** con dos entradas, **N** y **L**, que devuelva la lista que se obtiene al suprimir el elemento **N**-ésimo de la lista **L**.
  10. \*\* Programa **ADJUNTA**, con tres entradas, **N**, **X** y **L**, que incorpore **X** como elemento **N**-ésimo de la lista **L** (sin utilizar **INSERTA** –si es que se usa un dialecto de Logo que lo permita–).
-