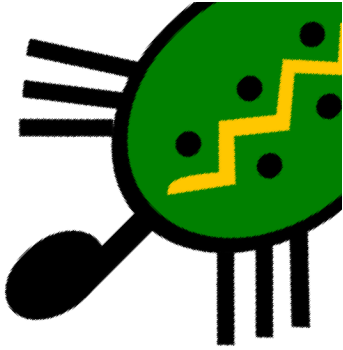


Capítulo 14

Modo multitortuga y Animación

14.1. Multitortuga

Se pueden tener varias tortugas activas en pantalla. Nada más iniciarse xLogo, sólo hay una tortuga disponible y su número es 0.

14.1.1. Las primitivas

Estas son las primitivas que se aplican al modo multitortuga:

Primitiva	Argumentos	Uso
`ponforma`, `pforma`	`número`	Puedes elegir tu tortuga preferida en la segunda etiqueta del menú **Herramientas** → **Preferencias**, pero también es posible con `ponforma`. El número `n` puede ir de 0 a 6. (0 es la forma triangular del Logo tradicional).
`forma`	`no`	Devuelve un número que representa la forma actual de la tortuga.
`pontortuga`, `ptortuga`	`número`	La tortuga número `n` es ahora la tortuga activa. Por defecto, cuando xLogo comienza, está activa la tortuga número 0.
`tortuga`	`no`	Da el número de la tortuga activa.
`tortugas`	`no`	Da una lista que contiene todos los números de tortuga actualmente en pantalla.
`eliminatortuga`	`número`	Elimina la tortuga número `n`
`ponmaximastortugas`, `pmt`	`número`	Fija el máximo número de tortugas
`maximastortugas`, `maxt`	`no`	Devuelve el máximo número de tortugas

Si quieres "crear" una nueva tortuga, puedes usar la primitiva `pontortuga` seguida del número de la nueva tortuga. Para evitar confusiones, la nueva tortuga se crea en el centro y es invisible (tienes que usar `muestratortuga` para verla). Así, la nueva tortuga es la activa, y será la que obedezca las clásicas primitivas mientras no cambies a otra tortuga con `pontortuga`.

El máximo número de tortugas disponibles también puede fijarse en el menú **Herramientas → Preferencias**.

14.1.2. Ejemplo. Curva de persecución

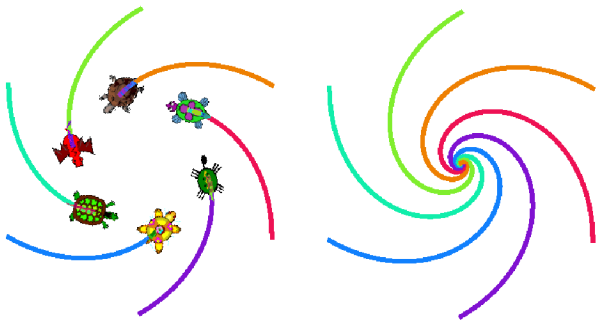
En este ejemplo vamos reproducir la **curva de persecución**. Vamos a distribuir `n` tortugas en los vértices de un polígono regular, y haremos que cada una se dirija hacia la posición de la tortuga situada a su derecha.

Como queremos ver a todas las tortugas en movimiento, vamos a hacer *un poco de trampa* y utilizar la primitiva `animacion` antes de explicarla. El motivo: al moverse las tortugas, la velocidad de refresco de la imagen no alcanza la velocidad de movimiento de las tortugas, y se genera un parpadeo bastante molesto:

```
para empieza :n
  borrapantalla ocultatortuga subelapiz
  animacion                              # Cierto, todavia no la explicamos
  inicio :n
  mientras [(distancia [0 0]) > 2 ]           # Funciona hasta que "chocan"
    [ repite :n
        [ pontortuga cuentarepite
          haz "mipos pos                      # Miramos donde esta la tortuga n
          pontortuga cuentarepite+1
          si cuentarepite+1>:n [pontortuga 1]
          ponrumbo hacia :mipos               # Orientamos la tortuga n +1
          avanza 2]
      refresca ]                              # Hacemos visibles los trazos
  detieneanimacion
  repite :n [
    pontortuga cuentarepite
    ocultatortuga ]          # Ocultamos todas las tortugas en la imagen final
fin

para inicio :n
  si :n <2 [escribe [Necesitas mas de una tortuga!] alto] # Control de error
  repite :n
```

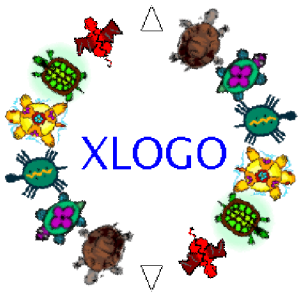
```
    [ pontortuga cuentarepite                          # Activamos tortuga
      ponforma cuentarepite                            # Cambiamos el tipo de tortuga
      muestratortuga                                   # Mostramos tortuga
      pongrosor 6
      haz "angulo cuentarepite*360/:n
      poncolorlapiz angcol :angulo subelapiz     # El color depende de la ubicacion
      ponxy (190*sen :angulo) (190*cos :angulo) bajalapiz]
fin

para angcol :x
  haz "r 127.5 *(1 + sen (:x))
  haz "g 127.5 *(1 + sen (:x + 120))
  haz "b 127.5 *(1 + sen (:x + 220))
  devuelve frase lista :r :g :b
fin
```

Si ejecutamos `empieza 6`, iremos viendo cómo las seis tortugas (cada una con una forma) van acercándose entre ellas, hasta juntarse en el centro. En ese momento, desparecen todas (la imagen de todas ellas superpuestas no es muy... *elegante*).

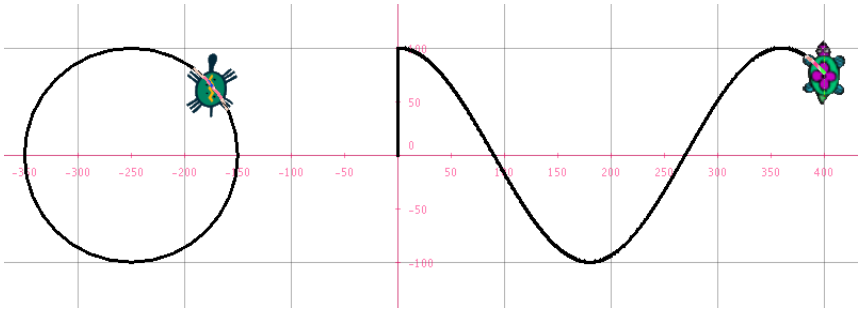


14.2. Ejercicios

1. Plantea un procedimiento que dibuje la "rosca" que aparece en la portada:



2. **Este problema implica conocimientos de Física**: Plantea un procedimiento que represente la muchas veces explicada analogía entre movimiento circular uniforme y movimiento armónico simple:



Para ello, necesitaremos tres tortugas:

a) Una que describa el movimiento circular

b) Otra que represente el paso del tiempo

c) Una tercera que describa el movimiento armónico

La primera tortuga irá trazando una circunferencia a la par que la segunda avanza un pequeño número de pasos (debe ajustarse para que se vea bien el movimiento) y la tercera se desplaza de modo que:

- Su ordenada coincida con la de la tortuga 1
- Su abscisa coincida con la de la tortuga 2

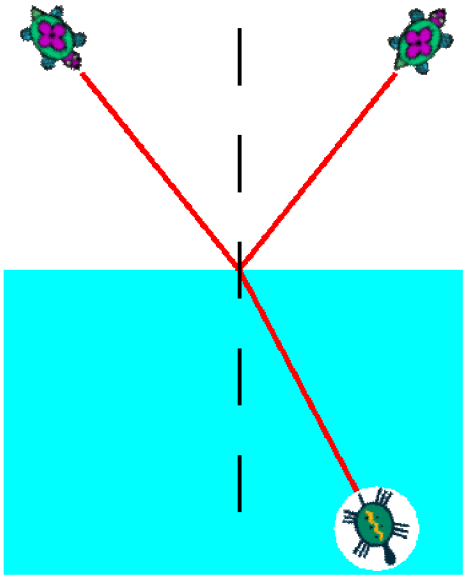
3. **Este problema implica conocimientos de Física**: Plantea un procedimiento que simule la reflexión y refracción de la luz. Para ello, necesitaremos dos tortugas, una para cada medio.

   Debemos trazar una línea que represente la separación de medios, pudiendo colorear al menos uno para distinguirlos bien. Una tortuga se desplazará desde un punto (que puede dejarse como argumento) hasta el origen (orientada correctamente con `ponrumbo`) y allí

   - Ella se "reflejará", invirtiendo su desplazamiento vertical
   - Apareceá la segunda tortuga que modificará su desplazamiento horizontal

   en ambos casos, siguiendo las leyes de Snell:

$$\begin{aligned} \text{Reflexión:} \quad & \alpha_{\text{inc}} = \alpha_{\text{refl}} \\ \text{Refracción:} \quad & n_{\text{inc}} \cdot \operatorname{sen} \alpha_{\text{inc}} = n_{\text{refr}} \cdot \operatorname{sen} \alpha_{\text{refr}} \end{aligned}$$



Puede "mejorarse" cambiando la forma de la tortuga para distinguir los dos "haces"

4. **Este problema implica conocimientos de Física**: Plantea un procedimiento que simule las fuerzas electrostáticas y gravitatorias de un sistema de masas o de cargas. Para ello:

   a) Ubicará n tortugas en posiciones que se indicarán como argumento

   b) Ubique otra tortuga con el ratón, con una "forma" distinta

   c) Determine vectorialmente la fuerza total en ese punto, y se desplace 3 ó 4 pasos en esa dirección

$$\vec{F} = K \cdot \frac{Q \cdot q}{r^2} \vec{u}_r$$

   d) Se detenga cuando la distancia a una de las tortugas sea, por ejemplo, 10

	Estás comprobando cómo no es "demasiado" complicado realizar simulaciones de Física con xLogo. Por supuesto, con esto entramos en temas que requieren conocimientos "avanzados" de Ciencias, pero podemos plantearnos usarlos en lugar de buscar software específico para cada tema. Como ves, las posibilidades son "infinitas".

14.3. Aplicación didáctica: lanzamiento de dos dados

Cuando se lanzan dos dados y se calcula la suma de los puntos de cada uno de ellos, se obtiene un resultado comprendido entre 2 y 12. En esta actividad vamos ver la distribución de frecuencias de las distintas tiradas y a representarla en un sencillo gráfico.

14.3.1. Simular el lanzamiento de un dado

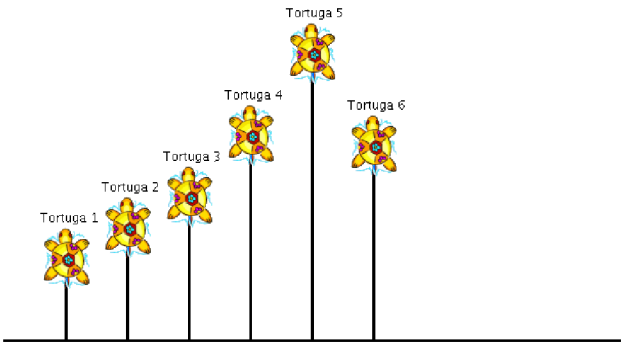
Para simular el lanzamiento de un dado, vamos a utilizar la primitiva `azar` (sección 7.3). `azar 6`, devuelve un número aleatorio comprendido entre 0 y 5. Por tanto, `(azar 6) + 1` devuelve una cantidad elegida aleatoriamente del conjunto `{1,2,3,4,5,6}`.

Como ya explicamos (página 64), debemos utilizar paréntesis; si no, XLOGO leería `azar 7`. Para evitar los paréntesis, se puede escribir `1 + azar 6`. Se define así el procedimiento `lanzar`, que simula el lanzamiento de un dado.

```
para lanzar
  devuelve 1 + azar 6
fin
```

14.3.2. El programa

Vamos a utilizar el modo *multitortuga* que acabamos de explicar, para así disponer de varias tortugas sobre la pantalla. La imagen siguiente nos indica qué queremos conseguir:



El objetivo es que cada tortuga, numerada de 2 a 12, avance un *paso de tortuga* cuando el resultado de la suma de la tirada de los dos dados coincida con su número. Por ejemplo, si la tirada de dados suma 8, la tortuga número 8 avanzará un paso.

La separación horizontal entre las tortugas es de 30 pasos de tortuga, y se colocarán a las tortugas con ayuda de los datos.

- Se colocará a la tortuga número 2 en `(-150 ; 0)`
- Se colocará a la tortuga número 3 en `(-120 ; 0)`
- Se colocará a la tortuga número 4 en `(-90 ; 0)`
- Se colocará a la tortuga número 5 en `(-60 ; 0)`
- ...

es decir:

```
pontortuga 2 ponpos [-150 0]
pontortuga 3 ponpos [-120 0]
pontortuga 4 ponpos [-90 0]
pontortuga 5 ponpos [-60 0]
pontortuga 6 ponpos [-30 0]
```

En lugar de copiar 11 veces prácticamente la misma línea de órdenes, usaremos `repitepara`, con la variable `:i` tomando los valores `2, 3, 4, ..., 12`.
Para colocar a las tortugas, creamos el procedimiento `inicia`

```
para inicia
  borrapantalla
  ocultatortuga
  repitepara [i 2 12]
   [ # coloca la tortuga
    pontortuga :i ponpos lista -150 + (:i - 2) * 30 0
     # escribe el numero de la tortuga justo debajo
    subelapiz retrocede 15
     rotula :i
    avanza 15 bajalapiz ]
fin
```

Observa la expresión `-150 + (:i - 2) * 30`. Con ello hacemos que el primer valor para la abscisa sea `-150`, y a cada nueva tortuga se añaden 30 (probar con distintos valores de `:i` si no se ve bien).

Finalmente, se obtiene el siguiente programa:

```
para lanzar
  devuelve 1 + azar 6
fin

para inicia
  borrapantalla
  ocultatortuga
  repitepara [i 2 12]
    [ # coloca la tortuga
     pontortuga :i ponpos lista -150 + (:i - 2)*30 0
      # escribe el numero de la tortuga justo debajo
     subelapiz retrocede 15
       rotula :i
     avanza 15 bajalapiz ]
fin
```

```
para empezar
  inicia
# Hacemos 1000 intentos
  repite 1000
    [ haz "suma lanzar+lanzar
      pontortuga :suma avanza 1 ]
 # indicamos las frecuencias de tirada
  repitepara [i 2 12]
    [ pontortuga :i
  # la ordenada de la tortuga representa el numero de tiradas
      hazlocal "frecuencia ultimo pos
      subelapiz avanza 10 giraizquierda 90
        avanza 10 giraderecha 90 bajalapiz
      rotula :frecuencia/1000*100 ]
fin
```

Veamos ahora una generalización de este programa. Aquí, se pedirán al usuario el número de dados deseados así como el número de lanzamientos a efectuar.

```
para lanzar :dados
  hazlocal "suma 0
  repite :dados
    [ hazlocal "suma :suma + 1 + azar 6 ]
  devuelve :suma
fin

para inicia
  borrapantalla ocultatortuga
  ponmaximastortugas :max + 1
  repitepara frase lista "i :min :max
   [ # coloca la tortuga
     pontortuga :i
     ponpos lista (:min - :max)/2*30 + (:i - :min)*30 0
     # escribe el numero de la tortuga justo debajo
     subelapiz retrocede 15
       rotula :i
     avanza 15 bajalapiz ]
fin

para empezar
  leeteclado [Numero de dados:] "dados
  si no numero? :dados
   [ es [largoetiqueta No es un numero!]
```

```
    alto ]
  haz "min :dados
  haz "max 6*:dados
  leeteclado [Numero de lanzamientos a realizar] "tiradas
  si no numero? :tiradas
   [ es [largoetiqueta El numero introducido no es valido!]
    alto ]
  inicia
# Debemos ajustar el paso para que no se salga de pantalla
  haz "paso :dados * 500/:tiradas
# Hacemos un numero de intentos igual a :tiradas
  repite :tiradas
    [ pontortuga lanzar :dados avanza :paso ]
# indicamos las frecuencias de tirada
  repitepara frase lista "i :min :max
    [ pontortuga :i
  # la ordenada de la tortuga representa el numero de tiradas
      hazlocal "frecuencia ultimo pos
  # normalizamos entre 0,1
      subelapiz avanza 10 giraizquierda 90
        avanza 10 giraderecha 90 bajalapiz
  # en caso de numeros grandes, los decimales son ... terribles
      rotula (redondea 10000*:frecuencia/:tiradas)/100 ]
fin
```

Piensa cómo podemos utilizar el sistema mostrado en el ejemplo para leer una lista de datos discretos y que las tortugas vayan representando un diagrama de barras con ellos. Pista: Deberás hacer más de una pasada a la lista, para determinar cuántos elementos distintos hay y ordenarlos de menor a mayor antes de releerla para hacer avanzar a las tortugas

14.4. Animación

Existen dos primitivas llamadas `animacion` y `refrescar` que permiten escribir órdenes sin que la tortuga las realice. `animacion` hace que la tortuga dibuje pero no lo muestre, es decir, a nuestros ojos no hace nada; al recibir la orden `refrescar` muestra todo el trabajo almacenado en memoria.

Primitivas	Uso
`animacion`	Se accede al modo de animación.

Primitivas	Uso
`detieneanimacion`	Detiene el modo animación, retornando al modo *normal*.
`refrescar`	En modo de animación, ejecuta las órdenes y actualiza la imagen

Mientras se escriben las órdenes en el modo de animación (una cámara de cine aparece a la izquierda del **Histórico de Comandos**), éstas no son ejecutadas en el **Área de Dibujo** sino que son almacenadas en memoria hasta que se introduce la orden `refrescar`.

Haciendo *clic* en este icono, se detiene el modo de animación, sin necesidad de usar la primitiva `detieneanimacion`.

Esto es muy útil para crear animaciones o conseguir que los dibujos se realicen rápidamente.

14.4.1. Ejemplo

Vamos a conseguir que un "camión" se desplace de izquierda a derecha de la pantalla. Empezamos por dibujar un camión (puedes cambiar el modelo, si este no te gusta) muy sencillo, un cuadrado como remolque, dos ruedas y una cabina simple:

```
para camion
  pongrosor 2 ocultatortuga
  repite 4
     [avanza 30 giraizquierda 90]
  giraderecha 90 avanza 15 giraizquierda 90 avanza 10
  giraizquierda 45 avanza (rc 2)*15 giraizquierda 135 avanza 25
  giraizquierda 90 goma avanza 5 ponlapiz circulo 5 rellena
  retrocede 20 goma retrocede 5 ponlapiz circulo 5 rellena
fin
```

y concluimos con la parte asociada a la animación:

```
para moviendose
  animacion
  repitepara [ lugar -300 +300 2]
```

```
   [ borrapantalla subelapiz ponx :lugar bajalapiz
     coche refrescar ]
  detieneanimacion
fin
```

El efecto final es el del camión desplazándose desde el punto `[-300 0]` hasta el `[300 0]`

Si aún tienes dudas, en la sección 18 mostraremos otra animación, esta vez sobre las cifras de la calculadora (cuenta atrás).

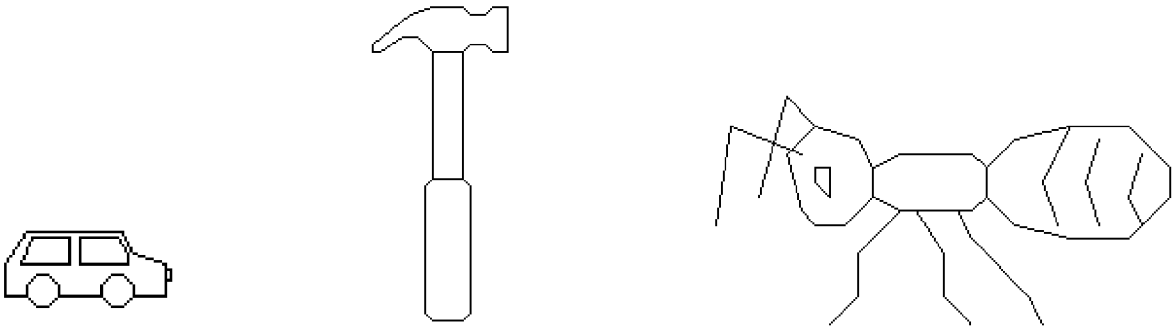
Este tipo de problemas pueden asociarse al estudio de los números complejos. Recordemos que la interpretación geométrica de un complejo permite sistematizar el análisis de:

- traslaciones
- rotaciones
- dilataciones

es decir, las **Homotecias**. En:

`http://neoparaiso.com/logo/numeros-complejos-aplicaciones.html`

muestran tres ejemplos de ello con las figuras de un coche, un martillo y una hormiga:



14.5. Ejercicios

Intenta reproducir con xLogo las tres animaciones propuestas:

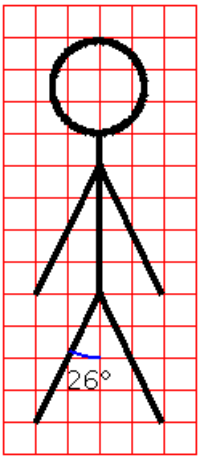
1. Desplazamiento de un coche
2. Giro de un martillo

3. Escalado de una hormiga

PERO usando correctamente las operaciones entre complejos.

Para representar los complejos con XLOGO deberás crear una lista con dos elementos, las coordenadas X e Y, y usar correctamente `primero` y `ultimo` para efectuar las operaciones.

14.6. El increible *monigote* creciente



En primer lugar, vamos a definir un procedimiento `monigote` que dibujará el monigote representado arriba, con un tamaño de nuestra elección.

```
para monigote :c
   giraizquierda 154 avanza 2.2*:c retrocede :c*2.2
   giraizquierda  52 avanza 2.2*:c retrocede :c*2.2
   giraizquierda 154 avanza 2*:c
   giraizquierda 154 avanza 2.2*:c retrocede :c*2.2
   giraizquierda  52 avanza 2.2*:c retrocede :c*2.2
   giraizquierda 154 avanza :c/2
   giraizquierda  90 repite 180
        [avanza :c/40 giraderecha 2]
   giraderecha 90
fin
```

Vamos ahora con la animación que creará la ilusión de que el monigote crece poco a poco. Para ello, escribimos `monigote 1`, despues `monigote 2` `monigote 3` ... hasta `monigote 75`. Entre cada trazado, se borrará la pantalla. Se obtienen los procedimientos siguientes:

```
para monigote :c
  si :c=75 [alto]
  giraizquierda 154 avanza 2.2*:c retrocede :c*2.2
```

```
  giraizquierda  52 avanza 2.2*:c retrocede :c*2.2
  giraizquierda 154 avanza 2*:c
  giraizquierda 154 avanza 2.2*:c retrocede :c*2.2
  giraizquierda  52 avanza 2.2*:c retrocede :c*2.2
  giraizquierda 154 avanza :c/2
  giraizquierda  90 repite 180
        [avanza :c/40 giraderecha 2]
  giraderecha 90
  borrapantalla ocultatortuga monigote :c+1
fin

para empezar
   borrapantalla ocultatortuga monigote 0
fin
```

Por último, para suavizar todo el proceso, vamos a servirnos del modo `animacion` y de la primitiva `refrescar`.

```
para monigote :c
  giraizquierda 154 avanza 2.2*:c retrocede :c*2.2
  giraizquierda  52 avanza 2.2*:c retrocede :c*2.2
  giraizquierda 154 avanza 2*:c
  giraizquierda 154 avanza 2.2*:c retrocede :c*2.2
  giraizquierda  52 avanza 2.2*:c retrocede :c*2.2
  giraizquierda 154 avanza :c/2
  giraizquierda  90 repite 180
        [avanza :c/40 giraderecha 2]
  giraderecha 90
  refresca
  borrapantalla ocultatortuga monigote :c+1
fin

para empezar
   borrapantalla ocultatortuga animacion
   monigote 0
   detieneanimacion
fin
```