TALLER DE GEOGEBRA 4

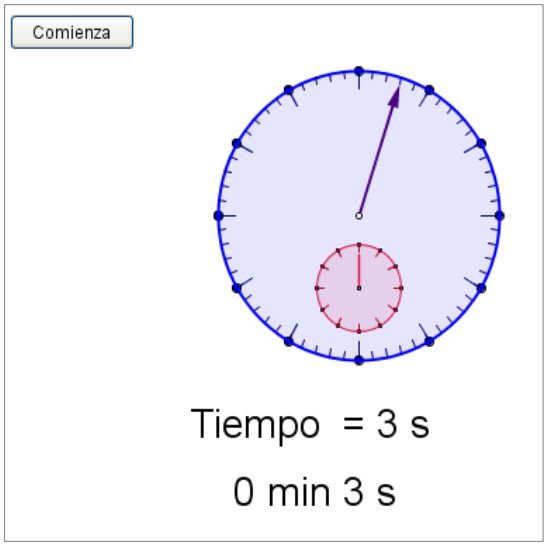
José Luis Álvarez

SEGUNDA PARTE: GEOGEBRA 4.0

P4. Cronómetro

Herramientas GG4: contador, comandos tiempo, texto dinámico.

Se trata de crear un cronómetro. La construcción tendrá una apariencia similar a la que se puede ver en la siguiente imagen:



Pasos de la construcción:

- ✓ En la barra de entrada escribimos, sucesivamente:

  t1= TomaTiempo[]

  t2=TomaTiempo[]

  El comando TomaTiempo recoge información del reloj del equipo local. Crea una lista que recoge, por orden: milisegundos, segundos, minutos, horas (de 0 a 23), fecha, mes, año, nombre del mes, nombre del día y día de la semana (1: domingo, 2:lunes…).

- ✓ Creamos el número t: t=0

- ✓ En las Propiedades de Objeto del número t, en la pestaña Deslizador fijamos los valores: Valor mínimo=0, Valor máximo=1, Incremento=0.001. También activamos Animación automática.

- ✓ En el Programa de Guión, en pestaña Al actualizar, escribimos:

  t2=TomaTiempo[].

  De ese modo la variable t2 se actualiza cada vez que cambia de valor el número t.

- ✓ En la barra de entrada escribimos: anima=false. De ese modo creamos el valor booleano anima que emplearemos más adelante.

- ✓ Convertimos el tiempo t1 a milisegundos. Para ello, en la barra de entrada escribimos:

  s1 = Elemento[t1, 1] + 1000 (Elemento[t1, 2]) + 60000 (Elemento[t1, 3]) + 3600000 (Elemento[t1, 4]) + 86400000 (Elemento[t1, 5])

- ✓ Convertimos el tiempo t2 a milisegundos. Para ello, en la barra de entrada escribimos:

  S2 = Elemento[t2, 1] + 1000 (Elemento[t2, 2]) + 60000 (Elemento[t2, 3]) + 3600000 (Elemento[t2, 4]) + 86400000 (Elemento[t2, 5])

- ✓ Calculamos el número tiempo, que medirá el tiempo transcurrido que vamos a medir con el cronómetro, expresado en segundos. En la barra de entrada escribimos:

  tiempo=(s2-s1)/1000

- ✓ Ahora vamos a construir los relojes. Creamos un punto A en la Vista Gráfica. Ese punto será el centro del cronómetro.
- ✓ Creamos el extremo de la aguja grande, que medirá los segundos. En la barra de entrada escribimos:

  B=rota[A+(0,1.9),- tiempo pi/30,A]

- ✓ Ahora creamos la aguja del cronómetro: en la barra de entrada escribimos

  vector[A,B]

- ✓ Creamos ahora la circunferencia del reloj:

  c=circunferencia[A,2]

- ✓ A continuación creamos las marcas del reloj. Primero marcas para señalar los segundos:

marcas1seg = Secuencia[Segmento[A + (1.9; k°), A + (2; k°)], k, 0, 360, 6]

- ✓ Ahora creamos las marcas para señalar intervalos de 5 segundos, algo mayores que las anteriores:

  marcas5seg = Secuencia[Segmento[A + (1.75; k°), A + (2; k°)], k, 0, 360, 30]

- ✓ Ahora vamos a crear un contador de minutos en el interior del anterior. Primero creamos el centro C de esta nueva circunferencia:

  C = A - (0, 1)

- ✓ Creamos el extremo de la aguja pequeña:

  D = Rota[C + (0, 0.45), -(tiempo π / 360), C]

- ✓ A continuación vamos a crear la aguja pequeña. Como es un segmento muy corto no empleamos la herramienta vector, pues la punta de la flecha sería muy grande en relación a la longitud de la aguja. Emplearemos la herramienta segmento. Escribimos:

  segmento[C, D]

- ✓ Creamos ahora la circunferencia del reloj:

  d=circunferencia[C,0.6]

- ✓ A continuación creamos las marcas del reloj. Primero marcas para señalar los segundos:

  marcas1min = Secuencia[Segmento[C + (0.5; k°), C + (0.6; k°)], k, 0, 360, 30]

- ✓ Ahora vamos a crear una variable que nos indique el número de minutos transcurrido. Para ello empleamos la función FLOOR:

  min = floor(tiempo / 60)

- ✓ Y ahora el resto, en segundos:

  seg = tiempo - min 60

- ✓ Ahora vamos a crear dos botones, uno para iniciar el cronómetro y el otro para pararlo.
- ✓ Seleccionamos la herramienta botón y hacemos clic en la parte superior derecha de la Vista Gráfica. Como subtítulo escribimos Comienza. Como Programa de guión escribimos:

  t1=TomaTiempo[]

  Valor[t2,t1]

  IniciaAnimación[true]

  anima=true

  El primer comando inicia la cuenta del tiempo, el segundo asigna el valor t1 a la variable t2. El tercer comando inicia la animación, con lo que empieza a variar t lo que hace que, a su vez, comience a variar t2 y, en consecuencia, empiece variar también la variable tiempo de la que depende el movimiento de las agujas que hemos creado. Por último, el cuarto comando cambia el estado del valor booleano anima del que va a depender que se vea el botón de parar o el de comenzar.

- ✓ En las Propiedades de Objeto del botón Comienza, en la pestaña Avanzado, en la casilla Condición para Exponer el Objeto escribimos

  anima==false

- ✓ Seleccionamos nuevamente la herramienta botón y hacemos clic en la parte superior derecha de la Vista Gráfica. Deberíamos colocarlo en el mismo lugar que el anterior, aunque podemos moverlo posteriormente. Como subtítulo escribimos Para. Como Programa de guión escribimos:

  IniciaAnimación[false]

  anima=false

  De este modo paramos la animación y devolvemos el valor false a la variable booleanas anima. Lo que conseguimos con ello es detener la aguja del cronómetro.

- ✓ En las Propiedades de Objeto del botón Para, en la pestaña Avanzado, en la casilla Condición para Exponer el Objeto escribimos

  anima

  De ese modo el botón es visible solamente cuando anima=true.

- ✓ Ahora creamos los textos que figuran bajo el cronómetro. Son los siguientes:

  texto1="Tiempo = " + tiempo + " s"

  texto2 = "" + min + " min " + seg + " s"

- ✓ Nos queda mejorar la estética de la construcción y fijar los objetos.

- ✓ Si lo deseamos, para mejorar la estética y facilitar la lectura podemos colocar marcas (puntos) sobre las circunferencias para señalizar mejor los intervalos de 5 en 5. En tal caso podemos escribir:

puntos5seg = Secuencia[Punto[c, t], t, 0, 1, 1/12]

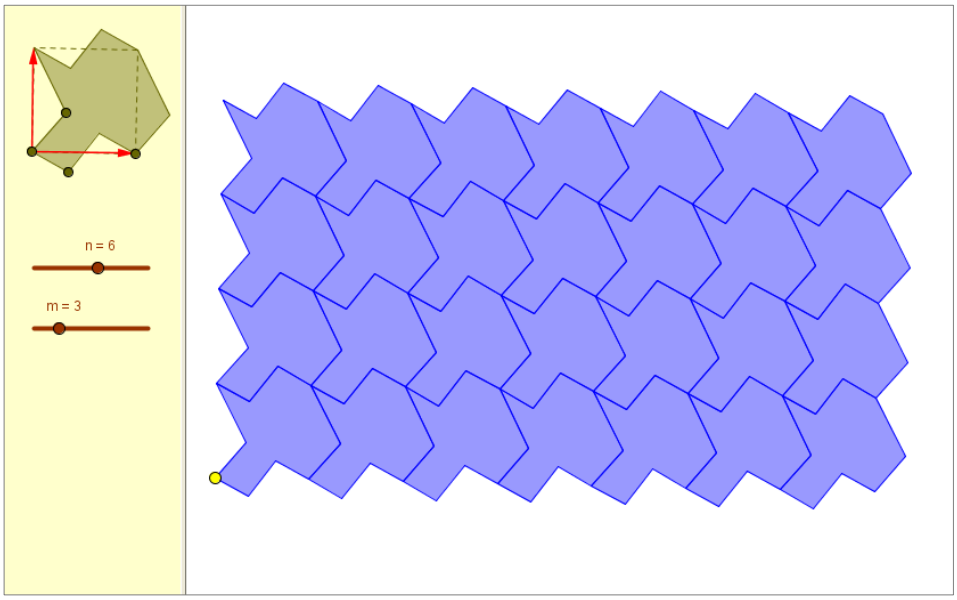
puntos5min = Secuencia[Punto[d, t], t, 0, 1, 1/12]

P5. Mosaico

Herramientas GG4: dos ventanas gráficas.

A partir de un cuadrado generamos la tesela que se resalta en la ventana de la izquierda. Rellenamos el plano a partir de un punto (señalado en amarillo en la imagen) controlando el número de teselas mediante dos deslizadores.

Destacamos la tesela a partir de la que se genera el mosaico, así como los deslizadores en otra ventana gráfica.



P6. Fractal

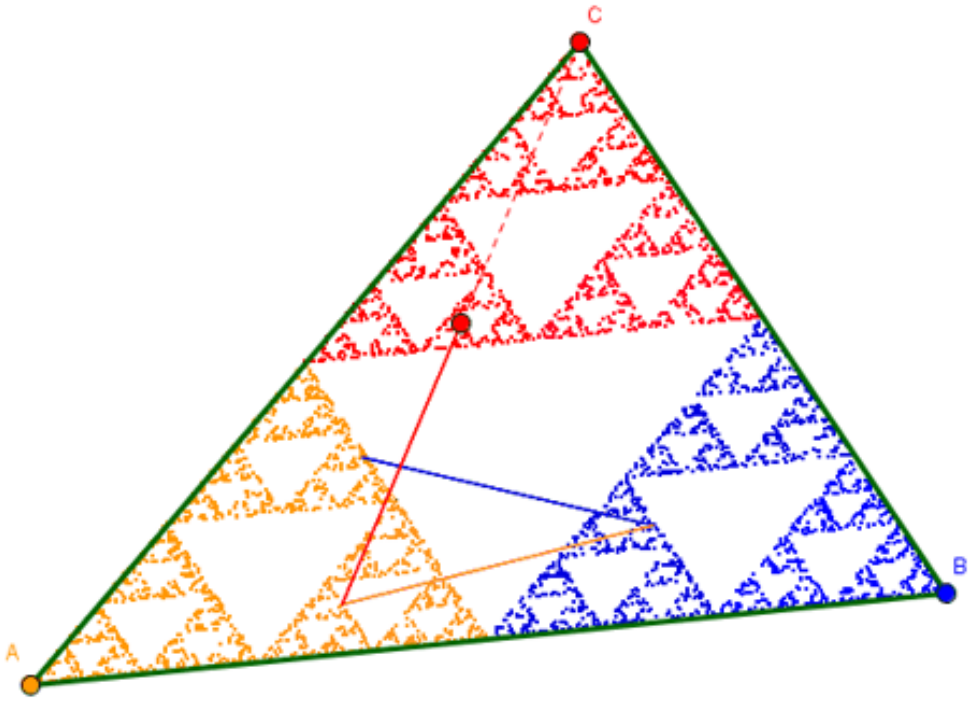
Herramientas GG4: botones, guión con Javascript.

Esta actividad forma parte del Curso Virtual de GeoGebra, que ofrece el ITE y cuyo autor es Rafael Losada.

Los fractales (como los generados con GeoGebra por Manuel Sada) son objetos matemáticos recientes y muy *atractivos*, tanto desde el punto de vista teórico como práctico. Con el nombre de "El juego del caos" (*Chaos game* o "algoritmo de iteración aleatoria") se conoce a un tipo de iteración que, a veces, genera un modelo fractal de forma no recursiva, como atractor de un sistema dinámico caótico.

Se eligen n puntos o vértices y otro punto D cualquiera. En cada paso, y aleatoriamente (con igual probabilidad 1/n), el punto D se dirige hacia uno de los n vértices, recorriendo solo una fracción constante de la distancia que los separa.

En este apartado, veremos cómo usar guiones GeoGebra y guiones JavaScript para generar el triángulo de Sierpinski mediante este proceso iterativo.



Protocolo de la construcción:

Nombre	Icono	Definición	Valor	Subtítulo
Punto A			A = (-1, -2)	
Punto B			B = (9, -1)	
Punto C			C = (5, 5)	
Triángulo tri		Polígono A, B, C	tri = 32	
Segmento a		Segmento [A, B] de Triángulo tri	a = 10.05	
Segmento b		Segmento [B, C] de Triángulo tri	b = 7.21	

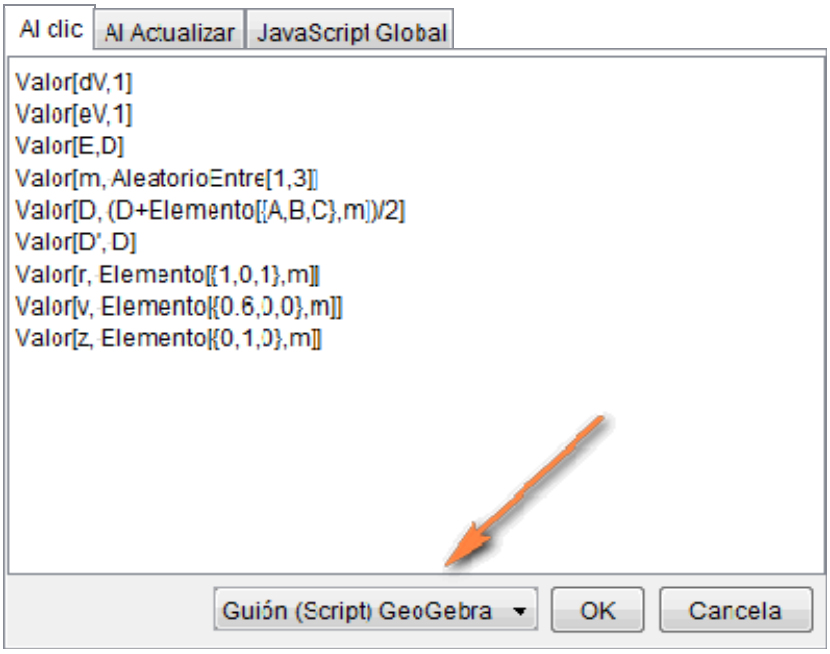
Segmento c		Segmento [C, A] de Triángulo tri	c = 9.22	
Número r			r = 0	
Número v			v = 0	
Número z			z= 0	
Número m			m = 0	
Valor Booleano dV			dV = true	
Valor Booleano eV			eV = true	
Punto D			D = (3, 0)	
Punto D'		D	D' = (3, 0)	
Punto E			E = (3, 0)	
Segmento d		Segmento [D, E]	d = 0	
Segmento e		Segmento [D, 2D - E]	e = 0	
Botón botón1			botón1	Una iteración
Botón botón2			botón2	Mil iteraciones
Botón botón3			botón3	Borrar

Construcción paso a paso:

- ✓ Creamos el triángulo ABC (cualquiera). Asignamos a A el color naranja, a B el color azul y a C el color rojo.
- ✓ Creamos las variables auxiliares `r = 0`, `v = 0`, `z = 0`, `m = 0`, `dV = true` y `eV = true`.
- ✓ Añadimos los puntos `D = (3, 0)` y `D' = D`.
- ✓ Minimizamos el tamaño del punto D y activamos su rastro. (El punto D' sirve para distinguir claramente su posición.)
- ✓ Añadimos el punto `E = (3, 0)` (que ocultamos) y los segmentos `d = Segmento [D, E]` y `e = Segmento [D, 2D - E]`.
- ✓ Asignamos a los puntos D y D' y a los segmentos **d** y **e** el color dinámico R: `r`, G: `v`, B: `z`.
- ✓ Asignamos a **d** y **e** la condición de visibilidad dV y eV, respectivamente, y a **e** un estilo discontinuo de trazo.
- ✓ Finalmente, añadimos tres botones, con los subtítulos que aparecen en el Protocolo.

Guiones GeoGebra

Solo queda asignar un guión (una serie de instrucciones) a cada pulsación de los botones. Usaremos guiones de GeoGebra para los botones 1 y 3, y un guión JavaScript para el botón 2.



Botón 1, "Una iteración":

Valor[dV,1] asigna a dV el valor 1 (true): el segmento d permanecerá visible.

Valor[eV,1] asigna a eV el valor 1 (true): el segmento e permanecerá visible.

Valor[E,D] asigna a E el valor de D: el punto E será la posición de partida de D.

Valor[m, AleatorioEntre[1,3]] asigna a m un entero aleatorio entre 1 y 3.

Valor[D, (D+Elemento[{A,B,C},m])/2] asigna a D el punto medio entre D y un vértice elegido al azar.

Valor[D', D] asigna a D' el mismo valor que tiene D.

Valor[r, Elemento[{1,0,1},m]] asigna al componente rojo un valor dependiente del vértice elegido.

Valor[v, Elemento[{0.6,0,0},m]] asigna al componente verde un valor dependiente del vértice elegido.

Valor[z, Elemento[{0,1,0},m]] asigna al componente azul un valor dependiente del vértice elegido.

Al acabar, debemos pulsar el botón .

Observemos que se usa el comando Valor en vez del signo =, debido a que, en un guión de GeoGebra, el signo = redefine el objeto, mientras que el comando Valor solo modifica el valor del objeto.

Botón 3, "Borra":

Valor[E,D] asigna a E el valor de D.

ZoomAleja[1] actualiza la vista gráfica, lo que provoca el borrado de los rastros.

Guiones JavaScript

Los guiones de GeoGebra son mucho más sencillos que los guiones JavaScript, pero estos últimos tienen mayor versatilidad y potencia.

En el botón 2, "Mil iteraciones", se hace uso de los comandos de JavaScript específicos de GeoGebra setValue, getXcoord, getYcoord y setCoords, así como de los comandos generales with, for y switch:

with(ggbApplet){ aplica el prefijo "ggbApplet." a todas las instrucciones siguientes.

setValue('dV',0); setValue('eV',0); asigna el valor 0 a dV y eV (oculta d y v).

x=getXcoord('D'); y=getYcoord('D'); recupera las coordenadas de D (x, y).

for(i=0; i<1000; i++){ Reitera mil veces.

n=1+Math.floor(3*Math.random()); crea un entero aleatorio entre 1 y 3.

switch(n){ según el valor de n, actualiza el valor de D y el color correspondiente.

case 1:

x=(getXcoord('A')+x)/2; y=(getYcoord('A')+y)/2;

setCoords('D', x,y); setValue('r', 1); setValue('v', 0.6); setValue('z', 0); break;

case 2:

x=(getXcoord('B')+x)/2; y=(getYcoord('B')+y)/2;

setCoords('D', x,y); setValue('r', 0); setValue('v', 0); setValue('z', 1); break;

case 3:

x=(getXcoord('C')+x)/2; y=(getYcoord('C')+y)/2;

setCoords('D', x,y); setValue('r', 1); setValue('v', 0); setValue('z', 0); break;

} cierra el switch, es decir, el selector según el valor de n.

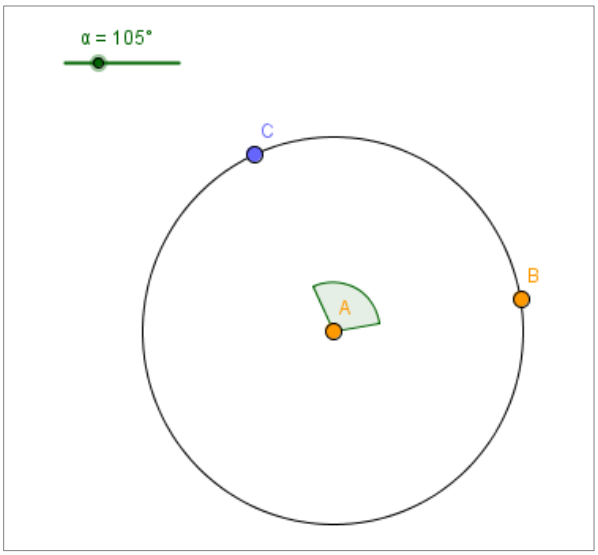
} cierra el bucle for que reitera mil veces.

} cierra la instrucción with.

P7. Gif animado

Herramientas GG4: exportar gif animado.

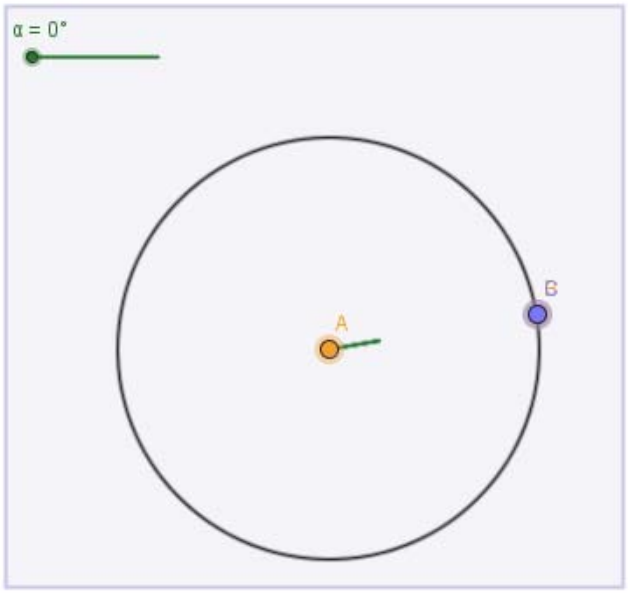
Se trata de crear una animación con GeoGebra (un punto que se mueve sobre una circunferencia) y exportar la construcción realizada como gif animado.



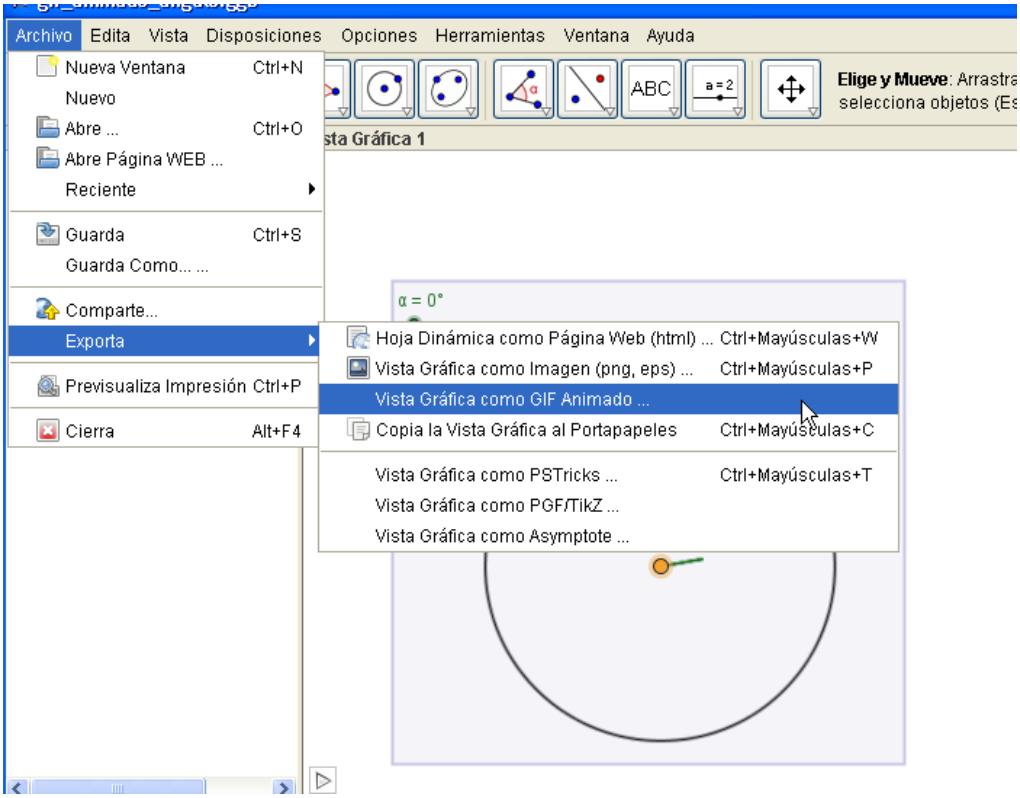
Los pasos de la construcción serían los siguientes:

- ✓ Creamos un deslizador, ángulo, con variación de 0 a 360º.
- ✓ Creamos una circunferencia, con centro en un punto A y que pasa por otro punto B.
- ✓ Creamos el punto C, al rotar B el ángulo dado por el deslizador, alrededor de A.
- ✓ Señalamos el ángulo BAC.
- ✓ Activamos la animación automática del deslizador.

Ahora debemos exportar la construcción como gif animado. Para ello seleccionamos la parte de la vista gráfica que formará parte del gif animado:



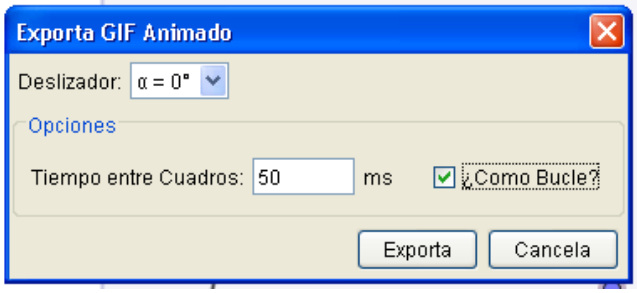
Una vez hecha la selección, en el menú Archivo elegimos la opción Exporta y, en la ventana emergente, seleccionamos Vista Gráfica como GIF animado:



Modificamos los parámetros del gif:

- ✓ si hay varios deslizadores, tendremos que elegir uno, dado que el gif animado no admite más que uno.
- ✓ La velocidad de la animación

✓ Marcamos la opción Bucle, si queremos que la animación sea continua.



Finalmente hacemos clic sobre el botón Exporta. E indicaremos el nombre y la carpeta de destino.

P8. Programación lineal

Herramientas GG4: inecuaciones, punto en objeto, hoja de cálculo, listas desplegables.

Se trata de resolver el siguiente problema de Programación Lineal, propuesto en una prueba de acceso a la universidad:

> Una empresa de instalaciones dispone de 195 kg de cobre, 20 kg de titanio y 14 kg de aluminio. Para fabricar 100 metros de cable del tipo A se necesitan 10 kg de cobre, 2 kg de titanio y 1 kg de alumnio. Para fabricar 100 metros de cable de tipo B se necesitan 15 kg de cobre, 1 kg de titanio y 1 kg de aluminio. El beneficio que obtiene la empresa por cada 100 metros de cable de tipo A fabricados es igual a 1500 euros, y por cada 100 metros de cable de tipo B es igual a 1000 euros.
>
> Calcúlense los metros de cable de cada tipo que han de fabricarse para maximizar el beneficio y determínese dicho beneficio máximo.
>
> Prueba de acceso a la universidad, Madrid, 2010

Con ayuda de una tabla estructuramos la información proporcionada por el enunciado:

	A	B	Disponible
Cobre	10	15	195
Titanio	2	1	20
Aluminio	1	1	14
Beneficio	1500	1000	

Podemos escribir las inecuaciones que nos permitirán delimitar la región factible y la función objetivo:

$$10x + 15y \leq 195$$

$$2x + y \leq 20$$

$$x + y \leq 14$$

$$x \geq 0$$

$$y \geq 0$$

La función objetivo es: f(x,y)=1500x+1000y

Escribimos una a una, en la barra de entrada, las inecuaciones y la función objetivo. Para las dos últimas inecuaciones escribiremos, respectivamente:

$$x + 0\,y \geq 0$$

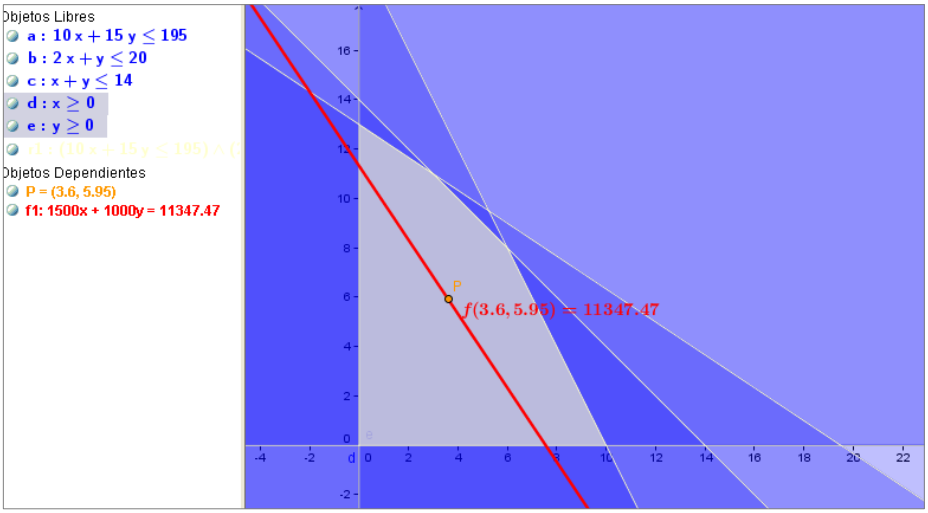
$$0\,x + y \geq 0$$

con el objeto de que GeoGebra considere que en ambos casos dependen de dos variables.

Definimos la región factible como la intersección de las cinco regiones representadas por las inecuaciones. Si son a, b, c, d y e los nombres asignados a dichas 5 regiones, en la barra de entrada escribimos:

g=a && b && c && d && e

Cambiamos colores y sombreados para destacar la región factible, de modo similar a como se ve en la siguiente imagen:



Creamos un punto en la región factible:

A=PuntoEn[g]

Calculamos el valor de la función objetivo en el punto A:

f(x(A),y(A))

Escribimos un texto en la pantalla

texto1="f("+x(A)+","+y(A)+") = "+f(x(A),y(A))

Situamos el texto en el punto A (Propiedades del objeto, Posición).

Moviendo el punto A por la región factible encontramos la solución del problema.

P9. Estudio estadístico sobre el Rácing de Santander

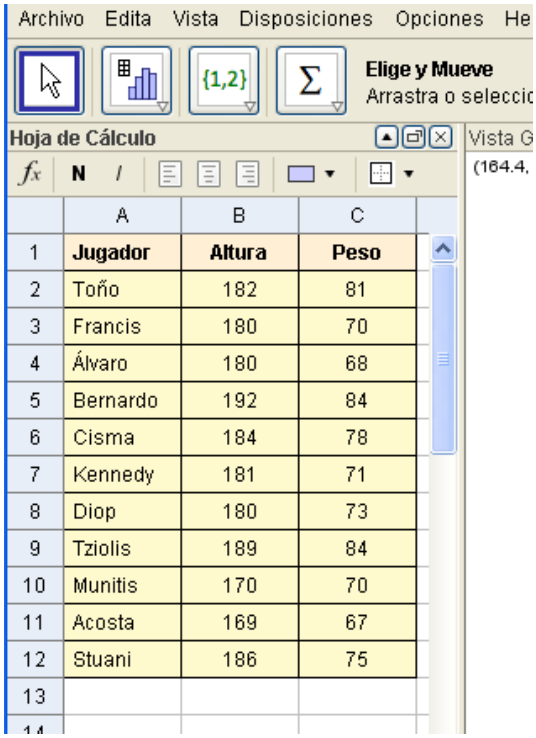
Herramientas GG4: Hoja de cálculo, análisis una variable, análisis de regresión.

Las alturas y pesos de la alineación titular del Rácing de Santander en un partido de esta temporada son los de la tabla siguiente:

Jugador	Toño	Francis	Álvaro	Bernardo	Cisma	Kennedy	Diop	Tziolis	Munitis	Acosta	Stuani
Altura	182	180	180	192	184	181	180	189	170	169	186
Peso	81	70	68	84	78	71	73	84	70	67	75

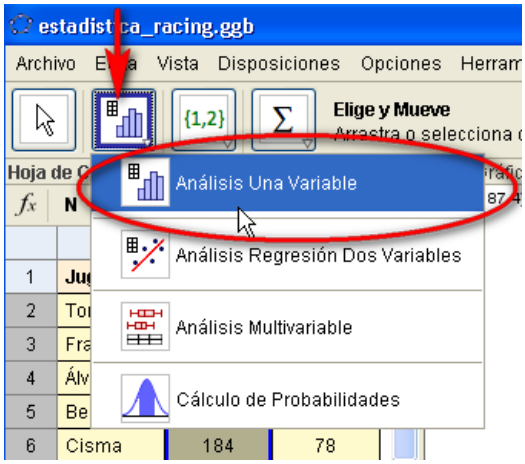
a) Elabora un estudio estadístico de las alturas de los jugadores.
b) Elabora un estudio estadístico de sus pesos.
c) Estudia la correlación existente entre las alturas y los pesos de los jugadores.
d) ¿Qué peso cabría esperar en un jugador de 190 cm de altura?

Para resolver el problema, introducimos los datos en la hoja de cálculo de GeoGebra:

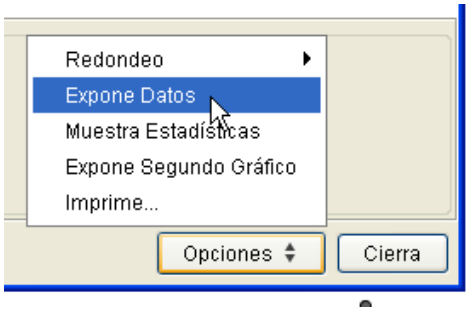


	A	B	C
1	Jugador	Altura	Peso
2	Toño	182	81
3	Francis	180	70
4	Álvaro	180	68
5	Bernardo	192	84
6	Cisma	184	78
7	Kennedy	181	71
8	Diop	180	73
9	Tziolis	189	84
10	Munitis	170	70
11	Acosta	169	67
12	Stuani	186	75
13			

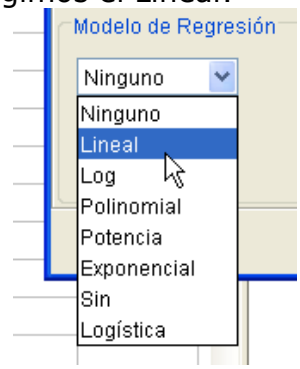
Para responder a la primera pregunta, seleccionamos las alturas (rango B2:B12) y seleccionamos la herramienta Análisis Una variable:

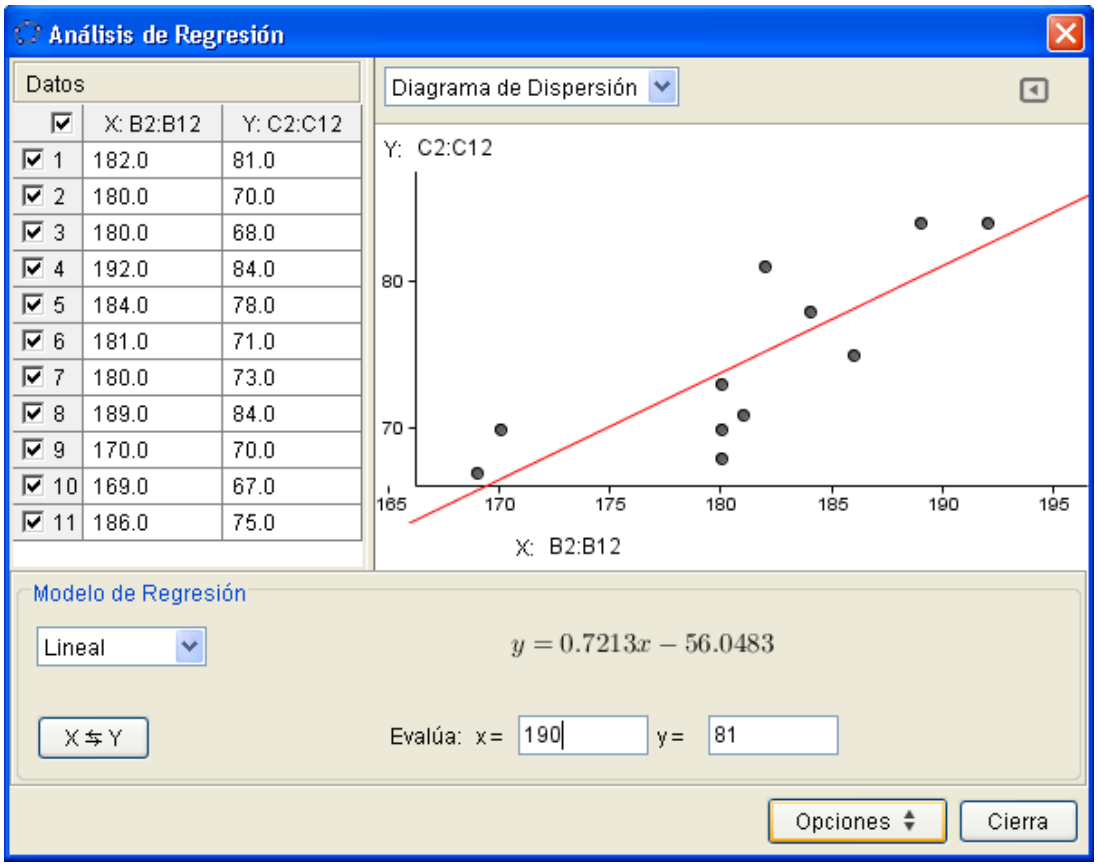


Para el análisis estadístico de los pesos, haremos lo mismo con el rango C2:C12. Para el estudio estadístico de la correlación, empleamos la herramienta Análisis Regresión Dos Variables. En la ventana emergente seleccionamos la opción Expone Datos:



Como modelo de regresión elegimos el Lineal:





P10. Distribución de tomates

Herramientas GG4: hoja de cálculo, cálculo de probabilidades.

Los diámetros de una determinada población de tomates siguen una distribución normal de 7,2 cm de media y 1,7 cm de desviación estándar.

a) Si el diámetro comercial mínimo es de 6 cm, ¿cuál es la proporción de tomates rechazados?
b) Si se quisiera aceptar el 90% de la población de tomates, ¿cuál debería ser el diámetro comercial mínimo?

En este caso utilizamos la herramienta Cálculo de Probabilidades: