

LISTADO DE PROYECTOS PROPUESTOS

ÓSCAR SÁNCHEZ ROMERO

ÍNDICE

1.	Proyectos básicos (01Basico)	2
1.1.	Hola Mundo (01HolaMundo.sb3)	2
1.2.	¿Me mueves? (02MeMueves.sb3)	2
1.3.	Pistas matemáticas (03MathClues.sb3)	2
1.4.	Pistas matemáticas versión 0 (04MathCluesJuego.sb3)	3
1.5.	Pistas matemáticas versión 1 (05MathCluesJuegoV1.sb3)	3
1.6.	Pistas matemáticas versión 2 (06MathCluesJuego.sb3)	3
1.7.	Cambio a base 2 (07PasarBase2Basico.sb3)	4
2.	Proyectos relacionados con el uso de ángulos (02Angulos)	4
2.1.	¿Sabes llegar? (01SabesLlegar.sb3)	4
2.2.	¿Sabes llegar? Versión juego (02SaberLlegarJuego.sb3)	5
2.3.	Suma de ángulos (03SumaAngulos.sb3)	5
2.4.	Acierta el ángulo Versión juego (04AciertaAnguloJuego.sb3)	6
3.	Letra del NIF (03NIF)	6
3.1.	Cálculo (mejorable) de la letra del NIF (01LetraNIFMejorable.sb3)	6
3.2.	Cálculo (mejorado) de la letra del NIF (02LetraNIFMejorado.sb3)	7
4.	Cronómetros (04Crono)	7
4.1.	Programación de un cronómetro (01Crono1Minuto.sb3)	7
4.2.	Programación de un temporizador (02Temporizador1Minuto.sb3)	8
5.	Pizarras matemáticas (05PizarrasMatematicas)	8
5.1.	Pizarra básica (01Pizarra.sb3)	8
5.2.	Pizarra simétrica (02PiezarraSimetrica.sb3)	8
5.3.	Pizarra dilatación (03PizarraDilatacion.sb3)	9
5.4.	Pizarra giro (04PizarraGiros.sb3)	9
6.	Representación de polígonos y círculos (06PoligonosYCirculos)	10
6.1.	Representación polígonos regulares (01Poligonos.sb3)	10
6.2.	Representación de círculos (02circulo.sb3)	10
6.3.	Coches de choque (03CochesDeChoque.sb3)	10
6.4.	Coches de choque versión juego (04CochesDeChoqueJuego.sb3)	11
7.	Funciones	11
7.1.	Evalúa una función (01EvaluaFuncionSimple.sb3)	11
7.2.	Evalúa una función (03GrafFuncion.sb3)	12
7.3.	Evalúa una función (04GrafFuncionx20.sb3)	12
8.	Trigonometría	12

Date: 24 de febrero de 2026.

8.1. RazonesTrigonometricas (RazonesTrigonometricas.sb2)	13
12. Recurrencia	13
12.1. Definición recurrente de la función factorial (01Factorial.sb3)	13
12.2. Curva y copo de nieve de Koch (02KochFractal.sb3)	13

En este documento se listan los proyectos propuestos en este repositorio mostrando su posible objetivo educativo e indicaciones básicas para su uso.

1. PROYECTOS BÁSICOS (01BASICO)

1.1. Hola Mundo (01HolaMundo.sb3).

- **Objetivo:** Introducir a usuarios en el funcionamiento básico de Scratch.
- **Conceptos computacionales:** Mostrar uso básico de bloques, objetos y fondos. Hacer interactuar objetos mediante el envío de mensajes.
- **Instrucciones para su uso:** Se ejecuta al pulsar la bandera verde.

1.2. ¿Me mueves? (02MeMueves.sb3).

- **Objetivo:** Control por parte del usuario del movimiento de un objeto mediante el uso de teclas o clics.
- **Investigación matemática:** Presentar el sistema de coordenadas que identifica la posición de un objeto. Observar la relevancia del atributo *dirección* del objeto y relacionarlo con ángulos de giro.
- **Conceptos computacionales:** Empleo de eventos para provocar el desplazamiento del objeto. Empleo de bucles para simular movimiento en el tiempo
- **Instrucciones para su uso:**
 1. Se ejecuta al pulsar la bandera verde.
 2. Se emplean las teclas de dirección para mover el objeto.
 3. Al hacer clic en el objeto simula un salto.

1.3. Pistas matemáticas (03MathClues.sb3).

- **Objetivo:** Idem anterior.
- **Investigación matemática:** Interpretar gráficamente el significado de números enteros con ayuda de un fondo en el que se ha representado una recta real. Ajuste del tamaño de paso para que los desplazamientos coincidan con valores reseñados de la recta.
- **Conceptos computacionales:** Idem anterior.
- **Instrucciones para su uso:**
 1. Se ejecuta al pulsar la bandera verde.
 2. Se emplean las teclas de dirección a derecha e izquierda para mover el objeto a las posiciones que indique el profesor: “Id a la posición -8”.

1.4. Pistas matemáticas versión 0 (04MathCluesJuego.sb3).

- **Objetivo:** Crear un juego simple que proponga ir a una posición dada por una resta de números enteros.
- **Investigación matemática:** Trabajar el cálculo mental y la interpretación geométrica de las operaciones con números enteros en la recta real.
- **Conceptos computacionales:** Uso de bucles y condicionales, uso de variables (sumandos, contadores, tiempo,...), realizar operaciones y comparaciones básicas con números, generar números aleatorios.
- **Instrucciones para su uso:**
 1. Al hacer clic en la bandera verde la medusa propone una posición a la que ir mediante una operación de substracción. Por ejemplo: Ve a la posición $-1 - -4$
 2. Desplazarse con las flechas a izda o derecha hasta el resultado de la operación propuesta y pulsar espacio. En el ejemplo anterior sería pulsar espacio una vez la medusa esté en la posición 3 de la recta.
 3. El código propone 10 operaciones, se contabilizan los aciertos y al final se muestra el tiempo que se ha tardado en todo el proceso.

1.5. Pistas matemáticas versión 1 (05MathCluesJuegoV1.sb3).

- **Objetivo:** Modificar el juego anterior para que que proponga ir a una posición dada por una resta de números enteros pero esta vez forzando a realizar la interpretación gráfica de dicha operación.
- **Investigación matemática:** Trabajar el cálculo mental, y la interpretación geométrica de las operaciones de números enteros.
- **Conceptos computacionales:** Idem anterior.
- **Instrucciones para su uso:**
 1. Al hacer clic en la bandera verde la medusa propone una posición a la que ir mediante una operación de substracción. Por ejemplo: Ve a la posición $-1 - -4$
 2. Desplazarse con las flechas a izda o derecha hasta la primera posición de la operación propuesta y pulsar espacio. En el ejemplo anterior sería pulsar espacio una vez la medusa esté en la posición -1 de la recta.
 3. Desplazarse con las flechas a izda o derecha hasta el resultado de la operación propuesta y pulsar espacio. En el ejemplo anterior sería pulsar espacio una vez la medusa esté en la posición 3 de la recta.
 4. El código propone 10 operaciones, se contabilizan los aciertos y al final se muestra el tiempo que se ha tardado en todo el proceso.

1.6. Pistas matemáticas versión 2 (06MathCluesJuego.sb3).

- **Objetivo:** Crear un juego simple que proponga ir a una posición dada por una resta de números enteros, donde las expresiones matemáticas tengan paréntesis.
- **Investigación matemática:** Trabajar el cálculo mental y la interpretación geométrica de la recta real.

- **Conceptos computacionales:** Idem anteriores. Operadores de manejo de cadenas de caracteres.

- **Instrucciones para su uso:**

1. Al hacer clic en la bandera verde la medusa propone una posición a la que ir mediante una operación de substracción. Por ejemplo: Ve a la posición $(-1) - (-4)$.
2. Desplazarse con las flechas a izda o derecha hasta el resultado de la operación propuesta y pulsar espacio. En el ejemplo anterior sería pulsar espacio una vez la medusa esté en la posición 3 de la recta.
3. El código propone 10 operaciones, se contabilizan los aciertos y al final se muestra el tiempo que se ha tardado en todo el proceso.

1.7. Cambio a base 2 (07PasarBase2Basico.sb3).

- **Objetivo:** Generar un código que nos proporcione la expresión en base 2 de un número natural cualquiera (inicialmente expresado en base 10).

- **Investigación matemática:** Trabajar la representación numérica en bases distintas a la decimal.

- **Conceptos computacionales:** Uso de operadores y de manejo de cadenas de caracteres.

- **Instrucciones para su uso:**

1. Ejecute la función ‘cambiar a base 2 el número natural’ introduciendo un valor natural y en la variable expresión aparecerá dicho número en base 2.
2. Es fácil generalizar este código para bases $b \leq 10$, mientras que para bases $b \leq 36$ es también factible empleando una cadena de caracteres de manera análoga a como se hace en el código ‘02LetraNIFMejorado.sb3’.

2. PROYECTOS RELACIONADOS CON EL USO DE ÁNGULOS (02ANGULOS)

2.1. ¿Sabes llegar? (01SabesLlegar.sb3).

- **Objetivo:** Proponer una actividad lúdica donde se ha de dirigir un objeto indicándole desplazamientos y giros evitando obstáculos.

- **Investigación matemática:** Empleo de ángulos y distancias para trazar la trayectoria de un objeto.

- **Conceptos computacionales:** Dirección del objeto y desplazamientos siguiendo esa orientación. Si se desea se puede cambiar a un nuevo fondo que puede ser generado mediante la utilidad de dibujo para tal fin.

- **Instrucciones para su uso:**

1. Una vez se hace clic el elefante solicita que se le desplace a lo largo del laberinto que simula el fondo.

2. Empleando sucesivos bloques de *mover* y *girar* indicando ángulos y distancias apropiadas conseguir una secuencia de movimientos que permitan al objeto pasar el laberinto.
3. Indicación: si se quiere visualizar la trayectoria seguida incluir un bloque *esperar* con un tiempo apropiado tras cada desplazamiento.

2.2. ¿Sabes llegar? Versión juego (02SaberLlegarJuego.sb3).

- **Objetivo:** Mejorar el proyecto anterior en dos sentidos. En primer lugar simular el movimiento del elefante al desplazarse y en segundo lugar incluir un premio para el elefante una vez ha llegado al final del laberinto.
- **Investigación matemática:** Empleo de ángulos y distancias para trazar la trayectoria de un objeto.
- **Conceptos computacionales:** Dirección del objeto y desplazamientos siguiendo esa orientación. Cambios de disfraz para simular movimiento. Uso de bucles. Creación de bloques propios. Establecer acciones tras el evento de que un objeto toque al otro.
- **Instrucciones para su uso:**
 1. Una vez se hace clic el elefante solicita que se le desplace a lo largo del laberinto que simula el fondo.
 2. Empleando sucesivos bloques de *mover* y *andar* indicando ángulos y distancias apropiadas conseguir una secuencia de movimientos que permitan al objeto pasar el laberinto.
 3. El bloque *andar* creado por el usuario ya simula el movimiento del elefante mediante cambio de disfraz y pausas apropiadas.
 4. Al llegar a los plátanos el elefante obtiene su recompensa.
 5. ¡Aviso! En el código del elefante se encuentra escondida la solución para pasar el laberinto para quien no tenga paciencia...

2.3. Suma de ángulos (03SumaAngulos.sb3).

- **Objetivo:** Demostración visual de que el resultado final de girar una suma de dos ángulos es equivalente a hacer consecutivamente ambos giros.
- **Investigación matemática:** Actividad para mostrar al estudiante el significado de la equivalencia de ángulos módulo 360 grados.
- **Conceptos computacionales:** Operaciones básicas con Scratch (operador módulo). Uso de variables y su manejo mediante desplazadores. Extensión de lápiz.
- **Instrucciones para su uso:**
 1. Al darle a la bandera verde uno de los escarabajos pide que se introduzcan los ángulos a sumar.
 2. Indicar un ángulos 1 y 2 moviendo los desplazadores.
 3. Pulsar espacio para que los escarabajos se giren, uno de ellos, consecutivamente según los ángulos 1 y 2 y el segundo la suma de ángulos.

2.4. Acierta el ángulo Versión juego (04AciertaAnguloJuego.sb3).

- **Objetivo:** Juego para que los usuarios identifiquen los ángulos básicos (de 30 en 30 grados).
- **Investigación matemática:** Reconocimiento de ángulos.
En el caso de abordar la programación de este proyecto hay que tener en cuenta que el atributo Dirección (D) de Scratch no sigue el mismo convenio que en matemáticas para identificar los ángulos mediante grados (G). Estos se relacionan mediante la transformación $G = (90 - D + 360) \text{ mód } 360$
- **Conceptos computacionales:** Uso de bucles y condicionales, variables y uso de funciones sencillas.
- **Instrucciones para su uso:**
 1. Al presionar la bandera verde la variable *Ángulo* tomará un valor entre 0 y 330.
 2. El gato empezará a girar a un ritmo regular.
 3. Presionar una tecla cualquiera cuando el gato esté orientado formando *Ángulo* con el eje X.

3. LETRA DEL NIF (03NIF)

El presente proyecto pretende demostrar cómo elementos de nuestra vida diaria, como puede ser la letra del NIF, son consecuencia de aplicar un algoritmo muy sencillo que involucra matemáticas. Este proceso, aunque suele ser desconocido, es simple: se calcula el resto de dividir el número del DNI entre 23 y al resultado (un número entero entre 0 y 22) se le asigna una letra mediante la siguiente tabla:

$0 \rightarrow T$	$1 \rightarrow R$	$2 \rightarrow W$	$3 \rightarrow A$	$4 \rightarrow G$	$5 \rightarrow M$	$6 \rightarrow Y$	$7 \rightarrow F$
$8 \rightarrow P$	$9 \rightarrow D$	$10 \rightarrow X$	$11 \rightarrow B$	$12 \rightarrow N$	$13 \rightarrow J$	$14 \rightarrow Z$	$15 \rightarrow S$
$16 \rightarrow Q$	$17 \rightarrow V$	$18 \rightarrow H$	$19 \rightarrow L$	$20 \rightarrow C$	$21 \rightarrow K$	$22 \rightarrow E$	

Así, por ejemplo, la letra asociada al DNI 23 000 000 sería T.

No obstante, presentamos dos versiones de programación de este algoritmo para, por un lado, convencer de que no hay una única manera de resolver un problema computacional, y por otro lado, incentivar a que siempre se busque aquella solución que sea lo más sencilla posible. Este proyecto se ha adaptado de uno análogo realizado en Octave disponible en el siguiente texto.

3.1. Cálculo (mejorable) de la letra del NIF (01LetraNIFMejorable.sb3).

- **Objetivo:** Mostrar una posible implementación del algoritmo para el cálculo del NIF mediante bucles y condicionales. Evidentemente, la selección del elemento dentro de la tabla de correspondencias será tediosa de programar.
- **Investigación matemática:** Adquirir conciencia de que la matemática está involucrada en nuestro día a día.
- **Conceptos computacionales:** Realizar preguntas y manejar la variable respuesta. Realizar operaciones básicas y manejo de variables. Manejo básico de cadenas de caracteres. Uso de condicionales.

■ Instrucciones para su uso:

1. Al hacer clic un perro nos pide un número de DNI,
2. una vez se le introduce, nos dice la letra que le corresponde en el NIF.

3.2. Cálculo (mejorado) de la letra del NIF (02LetraNIFMejorado.sb3).

- **Objetivo:** Mostrar una posible implementación del algoritmo para el cálculo del NIF mediante el manejo de cadenas de caracteres. Evidentemente, la selección de la letra del NIF dentro de una cadena de caracteres es bastante más eficiente y sencilla de implementar.
- **Investigación matemática:** Adquirir conciencia de que la matemática está involucrada en nuestro día a día.
- **Conceptos computacionales:** Realizar preguntas y manejar la variable respuesta. Realizar operaciones básicas y manejo de variables. Manejo de cadenas de caracteres.
- **Instrucciones para su uso:**
 1. Al hacer clic un perro nos pide un número de DNI,
 2. una vez se le introduce, nos dice la letra que le corresponde en el NIF.

4. CRONÓMETROS (04CRONO)

En esta sección se pretende trabajar con los segundos y minutos como variables sexagesimales a través de la implementación de un cronómetro (el tiempo avanza) y un temporizador (tiempo se descuenta). Ambos son proyectos adaptados de proyectos de José Ángel López en: vínculo cronómetro, vínculo temporizador.

4.1. Programación de un cronómetro (01Crono1Minuto.sb3).

- **Objetivo:** Programar un cronómetro digital empleando para ello sprites cuyos disfraces sean los dígitos del 0 al 9 para segundos, o bien del 0 al 5 para décimas de segundo.
- **Investigación matemática:** Trabajar la naturaleza sexagesimal de los segundos y minutos en nuestro sistema de medición temporal.
- **Conceptos computacionales:** Manejo de sprites, manejo de bucles.
- **Instrucciones para su uso:**
 1. Al pulsar en la bandera verde el cronómetro empieza a contar hasta que llega al tiempo final estimado.
 2. Una vez llegue al final, se puede proponer que el usuario programe cualquier acción llamativa: sonido, cambio de fondo, etc...

4.2. Programación de un temporizador (02Temporizador1Minuto.sb3).

- **Objetivo:** Programar un temporizador digital de 1 minuto empleando para sprites cuyos disfraces sean los dígitos del 0 al 9 para segundos, o bien del 0 al 5 para décimas de segundo. Para que sea más sencillo, la idea es alterar adecuadamente los disfraces para que al *pasar disfraz* el tiempo se cuente hacia atrás.
- **Investigación matemática:** Trabajar la naturaleza sexagesimal de los segundos y minutos en nuestro sistema de medición temporal.
- **Conceptos computacionales:** Manejo de sprites, manejo de bucles.
- **Instrucciones para su uso:**
 1. Al pulsar en la bandera verde el temporizador empieza a contar hasta que llega al tiempo final estimado.
 2. Una vez llegue al final, se puede proponer que el usuario programe cualquier acción llamativa: sonido, cambio de fondo, etc...

5. PIZARRAS MATEMÁTICAS (05PIZARRASMATEMATICAS)

Estos proyectos pretenden mostrar al usuario que en las alteraciones que hacemos a diario en imágenes digitales: zooms, giros, simetrías, etc... hay subyacentes simples transformaciones de coordenadas que los matemáticos denominan transformaciones en el plano. Para ello partimos de un proyecto básico donde se plantea programar una pizarra simple a partir de la extensión de *lápiz* que el usuario controla con su ratón. En siguientes versiones se añade al lápiz que controla el usuario otros objetos que simultáneamente están representando la misma imagen pero modificada de manera apropiada.

NOTA IMPORTANTE: Se ha detectado que en algunos equipos estos proyectos pueden no funcionar correctamente si no están en modo de pantalla completa, por lo que se recomienda su uso en cualquier prueba.

5.1. Pizarra básica (01Pizarra.sb3).

- **Objetivo:** Gracias al uso de la extensión de lápiz programar una pizarra que pinte el trazo que deja el ratón cuando este esté presionado.
- **Investigación matemática:**
- **Conceptos computacionales:** Uso de la extensión de lápiz, bucles, condicionales y el uso del sensor de *ratón presionado*.
- **Instrucciones para su uso:**
 1. Pasar a pantalla completa y hacer clic en la bandera verde.
 2. Al mover el ratón hacer clic cuando se quiera dibujar.
 3. Al presionar la tecla *b* se borra toda la pantalla.

5.2. Pizarra simétrica (02PiezarraSimetrica.sb3).

- **Objetivo:** Programar una pizarra que pinte el trazo que deja el ratón cuando este esté presionado y a la vez otro trazo simétrico al anterior.

- **Investigación matemática:** Observar el efecto de diversas simetrías en el plano:
 - con respecto al eje ordenadas: $(x, y) \rightarrow (x, -y)$,
 - con respecto al eje de abcisas: $(x, y) \rightarrow (-x, y)$,
 - con respecto a la diagonal principal: $(x, y) \rightarrow (-x, -y)$.
- **Conceptos computacionales:** Uso de la extensión de lápiz, bucles, condicionales y el uso del sensor de *ratón presionado*, y operaciones básicas a partir de las coordenadas de un objeto.
- **Instrucciones para su uso:**
 1. Pasar a pantalla completa y hacer clic en la bandera verde.
 2. Al mover el ratón hacer clic cuando se quiera dibujar.
 3. Al presionar la tecla *b* se borra toda la pantalla.

5.3. Pizarra dilatación (03PizarraDilatacion.sb3).

- **Objetivo:** Programar una pizarra que pinte el trazo que deja el ratón cuando este esté presionado y a la vez otro trazo dilatado con respecto al anterior.
- **Investigación matemática:** Observar el efecto de una dilatación (zoom) en el plano tomando el origen de coordenadas como punto central, esto es la aplicación $(x, y) \rightarrow \lambda(x, y)$.
- **Conceptos computacionales:** Uso de la extensión de lápiz, bucles, condicionales y el uso del sensor de *ratón presionado*, y operaciones básicas a partir de las coordenadas de un objeto.
- **Instrucciones para su uso:**
 1. Pasar a pantalla completa y hacer clic en la bandera verde.
 2. Seleccionar el valor de la dilatación (λ) y pulsar la tecla *p*.
 3. Al mover el ratón hacer clic cuando se quiera dibujar.
 4. Al presionar la tecla *b* se borra toda la pantalla.

5.4. Pizarra giro (04PizarraGiros.sb3).

- **Objetivo:** Programar una pizarra que pinte el trazo que deja el ratón cuando este esté presionado y a la vez otro trazo girado un cierto ángulo con respecto al anterior.
- **Investigación matemática:** Observar el efecto de un giro, θ , en el plano tomando el origen de coordenadas como punto central, esto es la aplicación:

$$\begin{pmatrix} x \\ y \end{pmatrix} \rightarrow \begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}.$$

- **Conceptos computacionales:** Uso de la extensión de lápiz, bucles, condicionales y el uso del sensor de *ratón presionado*, y operaciones básicas a partir de las coordenadas de un objeto.
- **Instrucciones para su uso:**
 1. Pasar a pantalla completa y hacer clic en la bandera verde.
 2. Seleccionar el valor del giro deseado (θ) y pulsar la tecla *p*.

3. Al mover el ratón hacer clic cuando se quiera dibujar.
4. Al presionar la tecla *b* se borra toda la pantalla.

6. REPRESENTACIÓN DE POLÍGONOS Y CÍRCULOS (06POLIGONOSYCÍRCULOS)

Estos proyectos plantean la representación gráfica con Scratch de polígonos regulares y círculos. Por último se añaden un par de proyectos denominados *Coches de choque* donde se muestra que los mismos comandos que nos permiten manejar un lápiz nos permitirían controlar un coche teledirigido.

6.1. Representación polígonos regulares (01Poligonos.sb3).

- **Objetivo:** Ser capaces de representar n-ágonos regulares, partiendo de los casos más sencillos.
- **Investigación matemática:** Las instrucciones para representar dichos polígonos se basan en conocer los ángulos que determinan sus lados adyacentes. El cálculo de dichos ángulos es un buen ejercicio de geometría básica ya que únicamente requiere conocer propiedades fundamentales de triángulos.
- **Conceptos computacionales:** Bucles, operaciones básicas con variables, creación de bloques, uso de la extensión de lápiz.
- **Instrucciones para su uso:**
 1. Este proyecto contiene las funciones básicas para representar triángulos, cuadrados y n-ágonos regulares con un número cualquiera de lados.
 2. Los comandos para hacer estas representaciones se deben ejecutar desde el propio código para invitar a la investigación.

6.2. Representación de círculos (02circulo.sb3).

- **Objetivo:** A partir del proyecto anterior convencerse de que una forma de tener una imagen muy próxima a una circunferencia en Scratch es realizar un n-ágono de 360 lados.
- **Investigación matemática:** Esta simple observación justifica que una circunferencia se puede ver como límite de polígonos (proceso de rectificación de la circunferencia).
- **Conceptos computacionales:** Bucles, uso de la extensión de lápiz.
- **Instrucciones para su uso:** Se ejecuta desde el propio código para invitar a la experimentación.

6.3. Coches de choque (03CochesDeChoque.sb3).

- **Objetivo:** Mostrar que un móvil se puede controlar mediante sus ángulos de giro y velocidad (distancia recorrida tras cada giro).

- **Investigación matemática:** giros y distancias son herramientas básicas para controlar la trayectoria de objetos (introducción a la robótica).
- **Conceptos computacionales:** Bucles y control de variables mediante teclas.
- **Instrucciones para su uso:**
 1. Al hacer clic en la bandera verde el coche empezará a avanzar girando según la variable *Ángulo* y avanzando a velocidad indicada por la variable *velocidad*.
 2. Con las flechas a derecha e izquierda se cambia de ángulo de giro.
 3. Con las flechas arriba y abajo se aumenta o disminuye la velocidad.
 4. El coche rebota si toca la pared.

6.4. Coches de choque versión juego (04CochesDeChoqueJuego.sb3).

- **Objetivo:** Conseguir un juego donde dos coches de choque tienen como objetivo golpear por detrás al otro coche y no ser golpeados.
- **Investigación matemática:** Idem anterior.
- **Conceptos computacionales:** Idem anterior
- **Instrucciones para su uso:**
 1. Al hacer clic en la bandera verde los coches empezarán a avanzar girando según la variable *Ángulo* y avanzando a velocidad indicada por la variable *velocidad*.
 2. El coche rojo se controla con las flechas según se indica en el proyecto anterior.
 3. El coche azul se controla con las teclas w, a, s, d emulando las flechas.
 4. Un coche consigue un punto si golpea por detrás al otro.

7. FUNCIONES

La posibilidad de crear nuevos bloques en Scratch nos posibilita el definir y trabajar con funciones matemáticas complejas, obtenidas a partir de las que vienen predefinidas. Sin embargo hay que señalar que en términos computacionales las funciones que permite declarar Scratch distan bastante de las de otros lenguajes, ya que las salidas que esta proporcionen han de ser reflejadas en modificaciones de variables globalmente definidas (ya que no tenemos habilitada la posibilidad de tener una variable únicamente declarada localmente dentro de una función). Sin embargo, las variables al ser declaradas si permiten seleccionar su alcance (un sprite o para todos).

7.1. Evalúa una función (01EvaluaFuncionSimple.sb3).

- **Objetivo:** Mostrar cómo se pueden definir funciones en el entorno de Scratch.
- **Investigación matemática:**
- **Conceptos computacionales:** Alcanzar familiaridad con uso variables, creación de bloques y definición de funciones complejas.
- **Instrucciones para su uso:**
 1. Determina la expresión de la función que pretendes evaluar en el bloque *Funcion*.

2. Haz click en la bandera, y sigue las instrucciones.
3. El valor que alcanza la función declarada en el punto indicado se va guardar en la variable *xEval* y si está dentro del entorno gráfico de Scratch $[-240, 240] \times [-180, 180]$ quedará representado.

7.2. Evalúa una función (03GrafFuncion.sb3).

- **Objetivo:** Representar gráficamente funciones en el entorno gráfico de Scratch por defecto (marco $[-240, 240] \times [-180, 180]$).
- **Investigación matemática:**
- **Conceptos computacionales:** Alcanzar familiaridad con uso variables, creación de bloques y definición de funciones complejas.
- **Instrucciones para su uso:**
 1. Determina la expresión de la función que pretendes evaluar en el bloque *Function*.
 2. Haz click en la bandera, y sigue las instrucciones.
 3. Quedarán representados los valores que la función tome en el entorno gráfico de Scratch por defecto, esto es, el rectángulo $[-240, 240] \times [-180, 180]$.

7.3. Evalúa una función (04GrafFuncionx20.sb3).

- **Objetivo:** Representar gráficamente funciones en el entorno gráfico de Scratch en un marco reescalado (marco $[-12, 12] \times [-9, 9]$).
- **Investigación matemática:** Cambio de variable, reescalamiento.
- **Conceptos computacionales:** Alcanzar familiaridad con uso variables, creación de bloques y definición de funciones complejas.
- **Instrucciones para su uso:**
 1. Determina la expresión de la función que pretendes evaluar en el bloque *Function*.
 2. Haz click en la bandera, y sigue las instrucciones.
 3. Quedarán representados los valores que la función tome en el entorno gráfico de Scratch reescalado que representa el fondo *Xy-grid-20px*, esto es, el rectángulo $[-12, 12] \times [-9, 9]$.

8. TRIGONOMETRÍA

Aunque el sistema de referencia que emplea Scratch para los ángulos no coincide con el que usamos los matemáticos, son muchas las posibilidades para trabajar con razones trigonométricas, tanto desde un punto de vista más teórico, como mostrando aplicaciones relacionadas con otras ciencias como el paralejo.

8.1. RazonesTrigonometricas (RazonesTrigonometricas.sb2).

- **Objetivo:** El objetivo principal de esta aplicación es mostrar al alumno o alumna como las razones trigonométricas sólo dependen de la amplitud del ángulo y no de las longitudes del triángulo elegido
- **Investigación matemática:** Idem anterior.
- **Conceptos computacionales:** Se puede usar este proyecto para que el alumno se familiarice con la visualización de variables.
- **Instrucciones para su uso:** Seleccione un valor del ángulo en el deslizador y pulse la bandera verde. Se puede observar que para un mismo ángulo:
 1. mientras el valor de la hipotenusa del triángulo va aumentando,
 2. los valores de las razones trigonométricas permanecen invariantes.
 3. Además, cuando el ángulo cambia, las razones también cambian.

12. RECURRENCIA

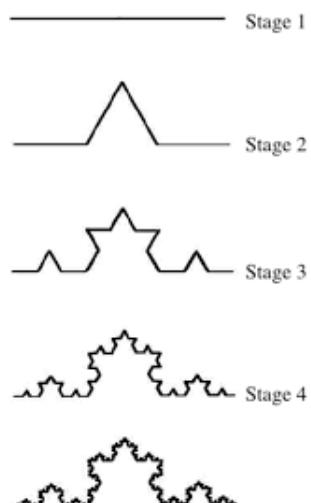
Por otro lado, la posibilidad de hacer llamadas recursivas entre funciones (que una función se llame a sí misma adoptando valores distintos para sus argumentos) nos permite familiarizar al estudiante con este concepto con raíces de marcado carácter matemático con ejercicios tan variopintos como la programación de la función factorial o bien la representación de aproximaciones de figuras fractales.

12.1. Definición recurrente de la función factorial (01Factorial.sb3).

- **Objetivo:** Implementar en Scratch la definición recurrente de la función factorial:
- $$n! = \begin{cases} \text{si } n = 1 \Rightarrow 1 \\ \text{si } n \geq 1 \Rightarrow n(n-1)! \end{cases} \quad n \in \mathbb{N} - \{0\}$$
- **Investigación matemática:** Función definida de forma recurrente.
 - **Conceptos computacionales:** Recursión computacional.
 - **Instrucciones para su uso:**
 1. Haz click en la bandera e introduce el número al que pretendes calcular el factorial.
 2. En el caso de que el número aportado sea es un natural positivo tendrás el valor de su factorial.

12.2. Curva y copo de nieve de Koch (02KochFractal.sb3).

- **Objetivo:**
- **Investigación matemática:** Visualizar el proceso recurrente de construcción de la curva y copos de Koch.
- **Conceptos computacionales:** Recursión computacional



■ Instrucciones para su uso:

1. Dependiendo del bloque de código que se ejecute bien obtendremos la curva de Koch o bien el copo de Koch (en la iteración recurrente indicada)