

# Programar: ¡Tan sencillo como hornear un pastel!

¿De qué se trata?

Después de esta unidad del curso sabrás qué son el lenguaje de programación y los controladores. ¡Conoces el concepto de algoritmo y ya estás muy bien preparado para tu primer intento de programación!

¿Todavía nunca programaste?

¡Ningún problema! Si alguna vez has cocinado u horneado algo siguiendo una receta, aprenderás a programar muy rápidamente.

**Ejercicio:**

**¡El orden es importante!**

Imagina que deseas hornear por primera vez un pastel de chocolate y cerezas. ¿Cómo lo haces?

Lo mejor es paso a paso:

- ① Buscas una receta
- ② Consigues los ingredientes
- ③ Mezcle los ingredientes en el orden indicado

¿Puedes poner en el orden correcto el resto de las instrucciones?

Escribe el número correspondiente antes del paso:

- Hornear el pastel en el horno
- Mezclar los ingredientes formando una
- masa

Precalentar el horno

¡El pastel ya está listo!

**Explicación: De la receta al programa**

La programación es exactamente lo mismo que una receta: se puede redactar qué pasos debe realizar un ordenador para resolver un problema.

Para ello el ordenador debe entender un «lenguaje de ordenador», en el que le das las instrucciones. Un lenguaje de esa clase se denomina lenguaje de programación.

Hay muchos lenguajes de programación. No todos los ordenadores pueden entender todos los lenguajes de programación.

**ft Controller**



El controlador fischertechnik que utilizarás es un tipo de miniordenador.

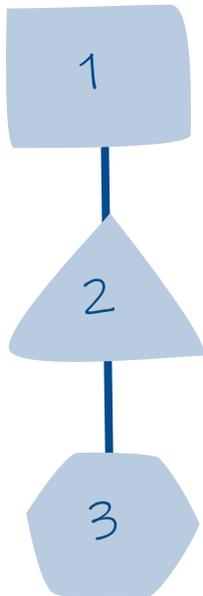
Entiende un lenguaje de programación que se llama «Scratch» y que es muy fácil de aprender.

**SCRATCH**

Muchas niñas, muchos niños y jóvenes en todo el mundo programan juegos, robots y distintos controladores con este lenguaje de programación.

¡No hay duda de que también lo «hablarás» muy rápido!

# Hablar como un profesional: Algoritmo



*Explicación: Algoritmo. ¿Qué significa eso?*

*Ahora conocerás un concepto importante que usarás más a menudo al programar: Algoritmo.*

*Se denomina algoritmo a una descripción precisa de los pasos que deben completarse para alcanzar un resultado determinado.*

*Una receta no es más que un algoritmo para cocinar o para hornear: Describe cómo se prepara un plato determinado. Para lograrlo no es necesario que hayas cocinado u horneado un pastel antes; si comprendes la receta y sigues las instrucciones, podrás hacerlo bien a la primera.*



*Piensa en un manual de uso, en la descripción de una ruta o en un manual de instrucciones: son todos algoritmos. Si se siguen las instrucciones un aparato se puede utilizar correctamente de inmediato, se encuentra la ruta buscada o un modelo en base a módulos se arma correctamente.*

*Con un (buen) algoritmo, por tanto, cualquiera puede conseguir (casi) cualquier cosa a la primera vez. El único requisito es seguir las instrucciones del algoritmo.*

*Para ello, naturalmente hay que entender y también poder realizar los pasos individuales, porque al menos una persona normalmente no puede ejecutar una instrucción como «ahora vuelas alrededor de la casa» o «ahora saltas sobre el río».*

*¿Y esto qué tiene que ver con un ordenador? Lo sabrás en la próxima página.*

# Hablar como un profesional: Algoritmos y ordenador

Del mismo modo en que se formulan algoritmos (=paso a paso, instrucciones) para una persona, también puede hacerse lo mismo para un ordenador.

Para eso hay que pensar en qué pasos se puede dividir una acción. Y un ordenador debe poder entender y ejecutar las instrucciones.

Por tanto, se puede redactar qué pasos debería realizar un ordenador para resolver un problema. Pero no se tiene que hacer en un lenguaje de programación. Un algoritmo se puede escribir también en alemán

o en español, y después se puede expresar en un lenguaje de programación.

Con la ayuda del algoritmo (= a la descripción de los pasos para la resolución) el programa se puede escribir con mucha facilidad precisamente en el lenguaje de programación que entiende el ordenador que se está utilizando.

La subdivisión en pasos y el orden correcto de los pasos ya se han considerado al redactar el algoritmo.

Completar aquí un ejemplo de algoritmo y programación analógica de Scratch.

## Modelo 1: Luz intermitente

¿De qué se trata? ¡Vas a construir un coche de policía con luz intermitente y a programarlo para que parpadee solo!



**1** ¡Pasa un coche de policía! La sirena es ruidosa y las luces en el techo parpadean. ¿Qué otros vehículos tienen luces intermitentes? Escribe o dibuja 2

**2** Tarea de construcción  
Construye un pequeño coche con luz intermitente. Usa para ello manual de instrucciones.

Consejo: La luz pequeña es una clase especial de lamparita y denomina LED. Un LED consume muy poca energía y no se calienta. Pero debes prestar atención a conectar los cables correctamente.



**3** Reflexiona: ¿Qué pasos debería realizar el «ordenador» para no solo encender una lámpara, sino también para que parpadee? Así se llega al algoritmo de una luz intermitente.

*Para recordar:*  
Un algoritmo es una descripción de instrucciones en el orden correcto, de manera que se alcance un resultado, por ej. un pastel horneado.

Se puede utilizar cualquier idioma y después pasar el algoritmo al lenguaje de programación correspondiente.

## Tu primer programa



**4** Encontrar instrucciones adecuadas para un programa de luz intermitente

Acordar un juego de roles: uno de vosotros se hace pasar por un ordenador. El ordenador tiene una luz conectada (una mano) y conoce la orden «encender». Cuando se le da la orden, la luz se enciende (él o ella levanta una mano). Reflexiona: ¿Qué instrucciones debe conocer, además, el «ordenador» para que también puedas enseñarle a hacer parpadear una lámpara?

Haz una lista de las instrucciones que el «ordenador» debe conocer para ello:

Encender,

---

---

---

### Redactar un programa

Redacta ahora un programa con esas instrucciones, de manera que tu «ordenador» (tu compañero/tu compañera) comience a «parpadear» (levantar la mano y volver a bajarla).

Prueba varias veces y discute el resultado: ¿El programa es correcto o aún falta algo?

Encender

### Para recordar:

Un programa es una serie de instrucciones. Esas instrucciones deben estar escritas en un lenguaje de programación que conozca y entienda el ordenador que debe ejecutar el programa.

## Tu primer programa Scratch

### 5 Comenzar con Scratch

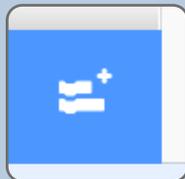
#### Preparación

1. Abre esta página web en el navegador:

<https://btsmart.ftscratch3.com>

Para que Scratch encuentre las órdenes para tu lámpara antes debes añadir la ampliación «BTSmart».

2. Para ello haz clic en ese pequeño campo abajo a la izquierda en la pantalla.

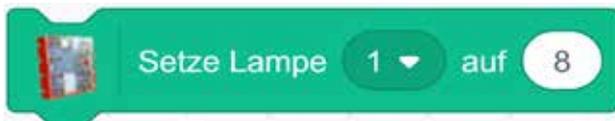


3. Selecciona la ampliación «BTSmart»:

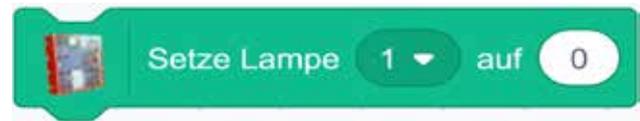


Selecciona la ampliación «BTSmart».

Necesitas varias órdenes de Scratch para tu luz intermitente. Encuéntralas en la lista de las órdenes de programación y arrástralas a tu área de programación:

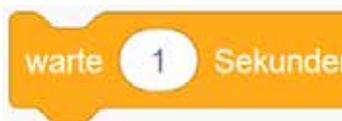


Lámpara encendida



Lámpara apagada

También necesitarás varias órdenes de Scratch desde el área control:



Espera 1 segundo, después pasa a la siguiente orden



Repetir de forma continua = repetir para siempre

Por último, necesitarás una orden de inicio, con la que comunicas al controlador que debe comenzar con el programa. El programa se inicia con esa orden cuando haces clic en la bandera verde pequeña arriba a la derecha:



## Tareas de programación

### 6 Programa tu luz intermitente con Scratch

Ahora ya no debería resultarte muy difícil programar tu primer programa Scratch «parpadear».

Antes de probar tu programa, controla que la lámpara (LED) esté conectada correctamente a la salida «M1» del controlador (conector rojo a la izquierda, conector verde a la derecha), y enciende la batería.

Conecta Scratch a tu controlador ft.

Inicia ahora tu programa. ¿Tu luz intermitente parpadea?

### 7 Guarda tu programa con el nombre «Luz intermitente»

Cambia ahora el nombre de tu proyecto de «Proyecto Scratch» a «Luz intermitente» y descárgalo en «Archivo» -> «Guardar en tu ordenador».



## Tarea experimental

### 8 Juegos intermitentes

Puedes regular la duración del parpadeo en la orden «Esperar 1 segundo».

Prueba varios otros patrones de parpadeo. Planifícalos antes en esta hoja.

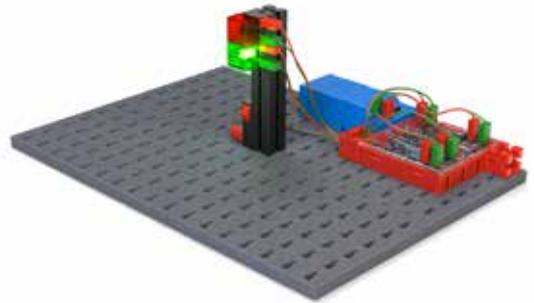
Ahora cambia el programa de modo que el parpadeo tenga lugar con un ritmo determinado.

¿Puedes lograr que tu LED parpadee coincidiendo con tu canción favorita?

<input type="checkbox"/>								
<input type="checkbox"/>								
<input type="checkbox"/>								

## Modelo 2: Cambio del semáforo

¿De qué se trata? En «rojo» detenerse, en «verde» circular ... Un semáforo regula el tráfico para que todas las calles puedan usarse o cruzarse de forma segura. Construyes tu propio semáforo para camino peatonal, que incluso se pone en «verde» al pulsar el botón.



**1**  Todos los semáforos contienen un componente similar al controlador ft. Cuenta: ¿Cuántos semáforos para peatones cruzas en tu camino a la escuela? ¿Cuántos tienen un botón para pulsar, para que cambie a verde? Pregunta a dos niñas o niños más y escribe aquí el resultado:

Nom- _____		_____	_____	De ellos, con botón:
Nombre: _____		_____	_____	
Nombre: _____		_____	_____	
Nombre: _____		_____	_____	

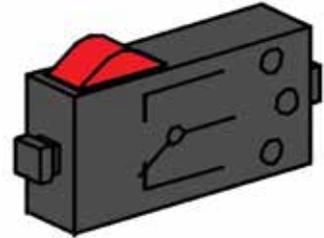
*Para recordar:*  
 Los pequeños miniordenadores que solo pueden hacer muy pocas cosas, y muy específicas, se llaman microcontroladores. Se encuentran en muchos objetos eléctricos cotidianos: Semáforos, barreras, un juguete eléctrico. A menudo tienen botones con los que puede iniciarse

**2** Escribe o dibuja:  
 ¿Qué dispositivos en tu hogar podrían tener microcontroladores?  
 Comenta tu selección con alguien de la clase.

# Un nuevo componente: Un botón pulsador

Con sensores, tu microcontrolador, el controlador ft, puede reaccionar a eventos en su entorno.

En esta tarea, conocerás el primer sensor: un pulsador. Los pulsadores funcionan solo con los estados «encendido» y «apagado». La diferencia con un interruptor reside en que el botón pulsador solo está encendido en tanto lo mantengas presionado. Si lo sueltas, el botón pulsador se apaga nuevamente por sí solo de inmediato.



El botón pulsador tiene un dibujo en el costado. Si ya estás un poco familiarizado con el tema de la energía y los circuitos eléctricos podrás reconocer qué conexiones debes utilizar. No obstante, en lo modelos en las propuestas de trabajo eso siempre figura en el manual de instrucciones o aquí en el dibujo.



Se lo conecta a una de las entradas «I1» a «I4» del controlador.

### 3 Tarea de construcción

Monta el semáforo para peatones como indica el manual de instrucciones.

### 4 Reflexiona: ¿Cómo podría verse el programa para el semáforo?

Debería indicar un ritmo fijo alternando «rojo» y «verde».

El LED rojo debería encenderse durante 2 segundos, el verde durante 1 segundo.

Haz nuevamente un juego de roles, en donde uno de vosotros represente el ordenador.

Vuestro programa:

Redacta el programa en vuestro lenguaje de programación sencillo y pruébalo con el «ordenador humano». Levanta un lápiz rojo y uno verde, para mostrar qué lámpara está encendida en ese momento. Cuando ninguna está encendida, las dos manos simplemente se mantienen abajo.

Prueba y mejora tu programa hasta que funcione bien.

<hr/> <hr/> <hr/> <hr/> <hr/> <hr/>	<hr/> <hr/> <hr/> <hr/> <hr/> <hr/>
-------------------------------------	-------------------------------------

# Tu programa de semáforo para peatones

## Repetición:

Para comenzar conecta nuevamente el controlador ft con Scratch

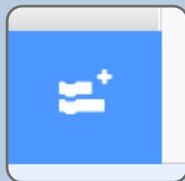
3. Selecciona la ampliación BTSmart:

1. Abre esta página web en el navegador:

<https://btsmart.ftscratch3.com>

Para que Scratch encuentre las órdenes para tu lámpara antes debes añadir la ampliación «BTSmart».

2. Para ello haz clic en ese pequeño campo abajo a la izquierda en la pantalla.



**5** Programa un cambio del semáforo en Scratch.

Para tu semáforo no necesitas ninguna nueva orden de Scratch. Ya has utilizado esta aquí:



Comienza con la misma programación que en el caso de la luz intermitente.

Intenta modificar la programación de manera que ambos LED se enciendan de forma alternada como en un semáforo automático.

Nota: El LED rojo debe estar conectado a «M1» y el LED verde a «M2». Nuevamente debes prestar atención a que el conector rojo se conecte a la izquierda, así como a «+».

Antes de iniciar el programa, no olvides encender la batería.

Inicia y prueba tu programa.

Guarda el programa con el nombre «Semáforo para peatones».

## Consejo

Es especialmente rápido si cargas tu programa «Luz intermitente», porque puedes reutilizar una parte del programa. Si no revisa nuevamente la hoja de soluciones para la luz intermitente.

# Solo verde si alguien quiere cruzar la calle: Semáforo a demanda

## 6 Semáforo a demanda

Ahora se complica un poco: El semáforo para peatones ahora solo debe cambiar a «verde» si alguien desea cruzar la calle. Un semáforo de esa clase se llama «semáforo a demanda».

En todos los lenguajes de programación hay órdenes para una condición. Esto significa que otras órdenes solo pueden realizarse si previamente ha sucedido algo determinado.

Para ello se encuentra en tu modelo el botón pulsador: Cuando éste es presionado el semáforo (rojo) debe pasar a «verde» después de transcurrido 1 segundo, y después de otro segundo nuevamente a «rojo».

Una orden para una condición indica por ej. si... entonces.

Primero escribe tu programa para un ordenador humano.

En nuestro ejemplo, por tanto: «Si se presiona el botón pulsador, entonces ...». Si se cumple con la condición se realizan las instrucciones subsiguientes; de lo contrario se omiten esas instrucciones.

Puedes usar las «órdenes» anteriores, pero ahora necesitas una nueva orden: La orden para una condición.

Ahora intenta redactar aquí una programación para el semáforo a demanda. Utiliza una condición (si ... botón pulsador) y prueba la programación con un «ordenador humano».

---

---

---

---

---

---

---

---

---

---

---

---

# Programación de Scratch semáforo a demanda

## 7 Programa tu programa Scratch «Semáforo a demanda»

Para ello necesitas dos elementos de Scratch que aún no has usado.

Observa qué colores tienen y los encontrarás con facilidad en las órdenes de Scratch.

Prueba: ¿Cómo se adaptan las órdenes?

La programación es especialmente rápida si cargas nuevamente tu programa «Semáforo para peatones», porque también aquí puedes reutilizar una gran parte del programa.

Inicia y prueba tu programa.

Guarda el programa terminado con el nombre «semáforo a demanda».



Una condición «Si ..., entonces» se expresa en Scratch un poco distinto.

Aquí se dice:  
En caso de ..., entonces

Esta es la verdadera condición que debe cumplirse:  
¿Está cerrado el interruptor?  
(= ¿está presionado el botón pulsador?)



# Tarea experimental: Semáforo para personas con discapacidad visual

## 8 Semáforo para personas con discapacidad visual

Muchos semáforos para peatones actualmente tienen una señal acústica para indicar a las personas con discapacidad visual o invidentes que el semáforo está en verde. Tam-

«Reproduce el sonido ... por ... golpes», de manera que el tono no sea desagradable y suene durante toda la fase en verde del semáforo.



bién se-  
má-  
fo-  
ro  
pue-  
de  
es-  
tar  
provis-  
to de  
una señal de

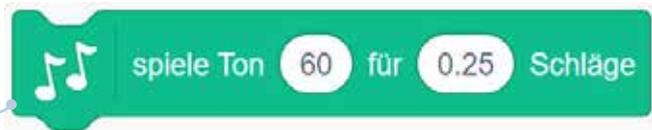
Para ello debes agregar la ampliación de Scratch añadir «Música».  
(Véase el procedimiento en la

Prueba tu programa.

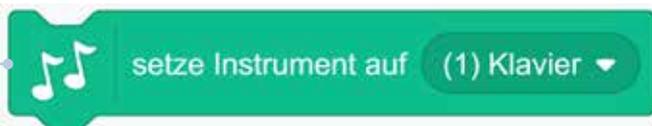
Guarda el programa con el nombre «Semáfo-  
ro para personas con discapacidad visual».

esa clase.

Esa ampliación contiene la orden *Reproduce el sonido ... por ... golpes*



Con la orden *Agrega un instrumento...*, puedes seleccionar otro instrumento al inicio de tu programa.

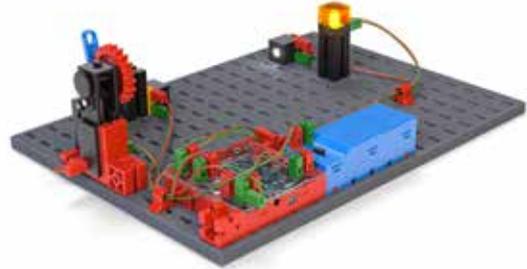


Cambia ahora tu programa «Semáforo a demanda», de modo que se escuche un sonido cuando el semáforo esté en «verde».

Ajusta el tono y la duración en el comando

## Modelo 3: Sistema de alarma

¿De qué se trata?  
 ¿Quieres asegurar un cofre del tesoro? ¿Tu habitación necesita un sistema de alarma?  
 ¡Construye ahora uno que realmente funciona!



**1** ¿Cómo un sistema de alarma en un cofre del tesoro podría percibir que éste puede ser abierto?

Alguien abre la tapa y, debido a eso, se suelta un pulsador presionado.

Alguien abre la tapa, debido a esto el cofre se tambalea, un sensor percibe el tambaleo.

Alguien abre la tapa y entra luz en la caja. Un sensor percibe la luz.

Alguien abre una tapa y, así, se presiona un pulsador.

Alguien abre la tapa y un sensor percibe un ruido.

Busca un compañero o una compañera y discute qué método le parece mejor. Márcalo con una cruz.

### Conocimiento especializado: Sensores

Para «percibir» algo, un ordenador en todos los casos debe estar conectado a un componente que pueda detectar información del entorno. Por tanto, por ejemplo, si está claro, si hay ruido, si está caliente, o si el piso está temblando.

Esos componentes se llaman sensores.

Hay sensores de muchas formas distintas y éstos «perciben», o mejor dicho, «miden» distintas cosas. Un sensor de luz reacciona a cambios de la luminosidad. Están incorporados en muchos dispositivos que conoces de tu vida diaria: Detectores de humo, teléfonos móviles, termómetros y muchos más.

## Un nuevo componente: Fototransistor

En este modelo, te presentamos otro sensor: Un fototransistor (módulo amarillo) asume aquí la función del pulsador.

El fototransistor es un sensor de luz que funciona como interruptor. Si percibe un nivel de luz suficiente, se enciende. Y si hay poca luz, se apaga.

Para el sistema de alarma se incorpora en una «barrera de luz». De este modo, un haz de luz enciende el fototransistor. Para que el haz de luz incida lo más concentrado posible en el punto correcto, el LED tiene adelante una pequeña lente y mantiene una tapa negra con un orificio. El fototransistor también es equipado con una tapa con orificios para que solo reaccione al haz de luz y no a la luz ambiente.

Si conectas la batería en tu modelo, deberías poder reconocer el punto de luz del LED en la tapa del fototransistor. El fototransistor ahora está «encendido». Si el haz de luz se interrumpe, el fototransistor se apaga.

¿Alguna vez has visto una película o una serie en donde algo muy valioso estaba protegido con barreras de luz? Ahora sabes bien cómo funciona.



**Ejemplo práctico:**  
Cuando el sol entra de lleno en la habitación, el fototransistor enciende el motor de una persiana enrollable. Si la persiana enrollable fue suficientemente bajada y la habitación ya está oscura, entonces el fototransistor detiene el motor para la cortina enrollable.

### **2** Tarea de construcción

Monta el sistema de alarma como indica el manual de instrucciones.

Si tu modelo está estructurado en base a la tarea anterior solo debes desmontar el semáforo; el resto puedes reutilizarlo.

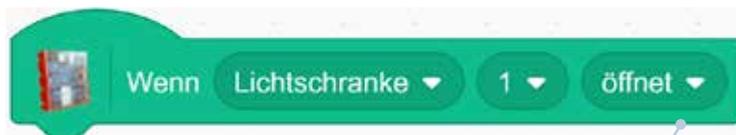
**Nota:** El fototransistor debe conectarse a «I1», de manera que el contacto marcado con rojo esté conectado a la conexión izquierda de la entrada en el controlador ft.

## Tareas de programación

### 3 Programa Scratch «Barrera de luz»

Escribe un programa Scratch que encienda el LED rojo cuando la barrera de luz esté interrumpida.

Para ello, puedes utilizar la siguiente orden de inicio:



En algunas órdenes de Scratch debes cambiar la orden de manera que se adapte a tu programa. Puedes ver si esto es posible en el pequeño símbolo triangular blanco. Presiona allí y busca la correcta.

Aquí debes cambiar la palabra «abre» a «cierra». El programa solo debe hacer algo si la barrera de luz se interrumpe.

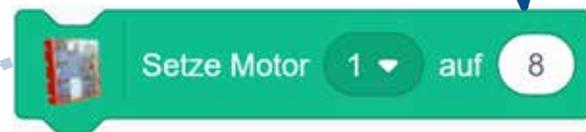
Guarda tu programa con el nombre «Barrera de luz».

## Tareas de programación

### 4 Programa Scratch «Sistema de alarma»

Ahora la barrera de luz debe convertirse en un sistema de alarma:  
Si la barrera de luz se interrumpe deben encenderse el LED rojo y el motor «ruidoso». Ambos también deben activarse cuando termine la interrupción de la barrera de luz.

Para el arranque del motor necesitas otra orden de Scratch desde la ampliación:  
Poner el motor ... en ...



Aquí cambias la velocidad del motor.

Prueba tu programa y guarda el programa con el nombre «Sistema de alarma».

### 5 Sistema de alarma con luz intermitente

Si se activa el sistema de alarma, el LED rojo ahora no solo debe encenderse, sino parpadear.

Modifica el programa Scratch de modo correspondiente y guárdalo con el nombre «Sistema de alarma con luz intermitente».

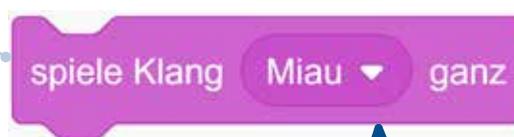
Consejo: Recuerda la luz intermitente en el modelo 1.

## Tarea experimental

### 6 Sistema de alarma con sirena

Scratch también puede emitir señales de alarma. La siguiente orden de Scratch reproduce un sonido que puedes escoger de una gran selección de distintos tonos:

Reproduce el sonido ... por completo



Aquí cambias el sonido. Pero: Solo si escogiste los sonidos antes. Cómo funciona eso se encuentra en el siguiente paso.

Más sonidos para elegir:  
En la pestaña «Timbres»  
puedes seleccionar un timbre adecuado haciendo clic en el siguiente símbolo:



Todos los sonidos que escoges aquí aparecen en tu orden de Scratch «reproduce sonido» en la lista de selección.

Reemplaza ahora en tu programa de Scratch «Sistema de alarma» de la tarea 4 el «motor ruidoso» por un sonido de alarma que se repita de forma continua.

Guarda el programa con el nombre «Sistema de alarma con sirena».

## Modelo 4: Barrera

¿De qué se trata? Muchos dispositivos eléctricos o juguetes en nuestra vida cotidiana también funcionan mecánicamente, es decir que los componentes se mueven. Con este modelo conocerás los principios importantes de esos dispositivos o juguetes.



### Introducción: Conocer las barreras como un profesional

Un elemento importante de los mecanismos que son accionados por un motor es la «detención de posición final»:

Si el mecanismo - por ej. una puerta de un garaje, una persiana enrollada o una barrera - llega al final del movimiento (a la «posición final»), el motor debe detenerse automáticamente.

Con esa finalidad se utilizan sensores que detectan si se ha alcanzado la posición final.

La barrera que construirás y controlarás en las siguientes tareas posee dos posiciones finales: completamente cerrada (horizontal) y completamente abierta (vertical).

Por lo tanto, necesita dos sensores que respectivamente detecten una posición final: dos botones pulsadores que se presionan si la barrera está completamente cerrada (abajo) o completamente abierta (arriba).

 **1** Control de término especializado: Explica a tu compañero o compañera qué son la «posición final» y la «detención de posición final».

La puerta de garaje y la barrera ya se mencionaron como ejemplos. ¿Se te ocurre algún otro?

### **2** Tarea de construcción

Monta la barrera como indica el manual de instrucciones. Si tu modelo del sistema de alarma de la tarea anterior aún está armado solo debes completar la barrera.

## Tareas de programación

### 3 Detención de posición final

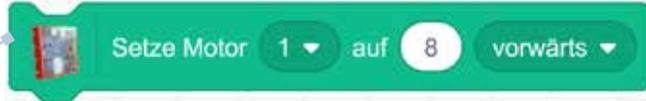
La barrera que primero se encuentra cerrada debe abrirse al inicio del programa y detenerse de forma automática tan pronto como se sitúe de forma vertical (detención de posición final).

Para la programación en Scratch necesitas tres nuevos elementos:

Espera hasta ...

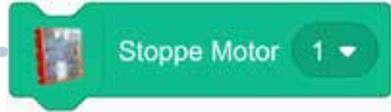
warte bis 

Poner el motor ... en ... hacia delante

Setze Motor 1 auf 8 vorwärts 

Con esta orden puedes seleccionar la dirección de la rotación del mo-

Detener el motor ...

Stoppe Motor 1 

Programa ahora tu algoritmo en Scratch y prueba el programa.

Regula la velocidad del motor de manera que la barrera no se mueva demasiado rápido ni demasiado lento.

Consejo: Si el motor rotara en la dirección equivocada, cambia entonces el conector rojo y el conector verde en el motor.

Guarda el programa con el nombre «Detención de posición final».

Si aún no tienes ninguna idea:

Describe el algoritmo primero con comandos sencillos para tu compañero o compañera y prueba si el algoritmo de tu «ordenador humano» se realiza correctamente.

## Tareas de programación

### 4) Abrir y cerrar la barrera

Después de la apertura la barrera debe permanecer abierta tres segundos y después debe cerrarse de nuevo.

Amplía ahora tu programa Scratch de la tarea 3 de modo correspondiente y pruébalo.

Consejo: El segundo botón pulsador está conectado en la entrada 2. Para abrir la barrera debes cambiar la dirección de rotación del motor («hacia atrás», en lugar de «hacia delante»).

Guarda el programa con el nombre «Barrera».

### 5) Barrera a demanda

Como en el caso del semáforo a demanda, ahora la barrera debe abrirse solo a demanda. En este caso, la barrera de luz debería asumir la función del botón pulsador: Tan pronto como un vehículo interrumpe la barrera de luz, la barrera debería abrirse y cerrarse nuevamente de forma automática después de 3 segundos.

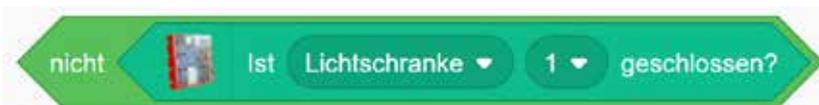
Amplía ahora tu programa Scratch de la tarea 3. En este caso necesitarás el siguiente operador que invierte el resultado

de la prueba:

Puesto que la barrera debería abrirse si la barrera está interrumpida, por tanto, no está cerrada.

Prueba el programa y guárdalo con el nombre «Barrera a demanda».

Consejo: Recuerda tu programa «sistema de alarma»



**Operador:**  
Las órdenes de color verde claro en Scratch se denominan operadores. Se los utiliza al comparar (por ej., ¿la temperatura es mayor o igual que hace 5 minutos?)  
O para probar si algo «no» es el caso.  
También hay órdenes para

### 6) Barrera a demanda con luz de advertencia

Para que ningún vehículo o peatón resulte lesionado al abrir o cerrar la barrera, el LED de advertencia rojo debería estar encendido cuando la barrera se mueve.

Completa tu programa Scratch y las órdenes necesarias para ello.

Guarda el programa con el nombre «Barrera a demanda con luz de advertencia».

## Tarea experimental

### 7 Barrera a demanda con advertencia de voz

En Scratch puedes hacer que tu ordenador emita textos.

Para ello, debes integrar la ampliación «Texto a voz»:



En esta ampliación encontrarás la orden



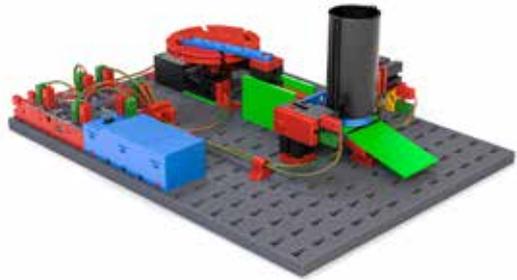
Di...

Amplía ahora tu programa Scratch de la tarea 6 de manera que, al abrirse la barrera suene la advertencia «Atención, se abre la barrera» y, al cerrarse la advertencia, «Atención, se cierra la barrera».

Guárdalo con el nombre «Barrera a demanda con advertencia de voz».

## Modelo 5: Contador de monedas

¿De qué se trata? ¿Un aparato que puede contar monedas? ¡Esto ofrece opciones muy diferentes para tu hucha!  
En este modelo, además, aprenderás cómo un ordenador puede «percibir» números.



### Introducción

En esta tarea propuesta le enseñarás a contar a tu controlador ft.  
Para ello tu programa Scratch debe percibir números.

En un lenguaje de programación eso funciona con así llamadas «variables».

Son «memorias» en las que puedes almacenar un valor. Y posteriormente puedes utilizarlo o emitirlo.

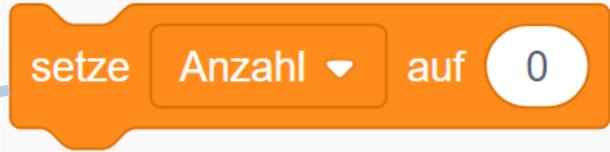
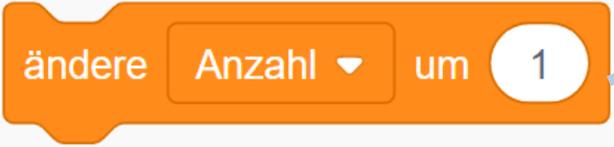
Debes comunicarle a Scratch que a partir de ahora deseas almacenar información y necesitas una variable.

### Una nueva variable

Con el botón «Nueva variable» genera una nueva memoria de esa clase, a la que le das un nombre conveniente (por ej. «Número»).

El valor de tus nuevas variables se muestra automáticamente en Scratch arriba a la izquierda en la «plataforma».

Establecer «mi variable» en 0 te permite empezar a contar con un valor inicial - aquí 0.

cambiar ... a ...  
con ello puedes agregar un valor.  
Para contar siempre se suma 1  
(cambia en 1 el número percibido)

### 2 Tarea de construcción

Monta el contador de monedas como indica el manual de instrucciones.

## Tareas de programación

### 3 Expulsión de monedas

El contador de monedas primero debe expulsar una moneda desde el depósito.

Para ello, afloja la varilla en el centro del plato giratorio (=el cubo) y primero gira con la mano el disco giratorio del contador de monedas.

Puedes observar que el módulo en el lado inferior del disco giratorio presiona el botón una vez en cada giro. Con ello, su programa después reconocerá que ha tenido lugar un giro. Aprieta ahora nuevamente el cubo del disco giratorio.

El mecanismo funciona de modo similar a la «detención de posición final» que ya has aprendido: Tan pronto como el botón pulsador esté presionado, el disco giratorio debe detenerse.

Programa la expulsión de una moneda desde el depósito en Scratch.

Prueba tu programa y guárdalo con el nombre «Expulsión de monedas».

### 4 Expulsión múltiple de monedas

Ahora deben expulsarse varias monedas de forma consecutiva.

Describe primero verbalmente el algoritmo con instrucciones simples.

Completa tu programa Scratch de la tarea 3 de modo correspondiente y pruébalo.

Atención: Probablemente el botón pulsador aún esté presionado cuando comiences con la siguiente expulsión de monedas. ¿Cómo puedes impedir que el programa se detenga de nuevo?

Guarda tu programa con el nombre «Expulsión múltiple de monedas».

## Tareas de programación

### 5 Expulsión de todas las monedas

Ahora la expulsión de monedas debe tener lugar hasta que ya no quede ninguna moneda en el depósito.

La barrera de luz te ayuda a detectar si aún están contenidas monedas en el depósito: Si se ha cerrado después de una expulsión de monedas (por tanto, no está interrumpida), el depósito está vacío.

Entonces ya no deben expulsarse más monedas.

Para la programación en Scratch necesitas ahora también la



también la

Programa la expulsión de monedas utilizando tu programa Scratch y prueba tu

### 6 Contador de monedas

Ahora puedes enseñar a tu programa a contar monedas.

Para ello, debes ampliar tu programa en una orden que al inicio fija en 0 el valor de tu variable («número») y después de la expulsión de monedas aumenta el valor en 1.

Amplía tu programa Scratch de modo correspondiente y pruébalo.

Guarda tu programa con el nombre «Contador de monedas».

## Tarea experimental

### **7** Contador de monedas con salida de voz

después de cada salida de monedas.

En la tarea propuesta de la barrera aprendiste cómo puedes emitir un texto como voz en Scratch.

Consejo: Utiliza tu variable («número») para la salida de voz.

Amplía tu programa Scratch de manera que cuente en voz alta las monedas que han salido, emitiendo el número actual

## Modelo 6: Un ventilador que te refresca

¿De qué se trata? ¿Cómo pueden reaccionar los dispositivos y los ordenadores a su entorno?

Los componentes se denominan sensores, de los que ya has conocido algunos.

Aquí se trata de que un ventilador, al aumentar el calor, gire más rápido y te refresque.



### Un sensor que mide el calor

En esta tarea conocerás otra clase de sensor.

Este sensor no «se enciende» y «se apaga», como un botón pulsador o como el fototransistor.

Este sensor puede medir algo, como la humedad o la temperatura. Y puede transferir esos valores al ordenador. Así puedes utilizarlos en la programación.

El componente que utilizamos aquí reacciona al calor. Pero de un modo particular: El valor medido no corresponde a la temperatura, sino que el valor medido se vuelve menor al aumentar el calor, en lugar de hacerse mayor.



Con eso nos basta para el modelo de ventilador en esta tarea propuesta. En la tarea experimental aprenderás cómo utilizarlo para determinar la temperatura.

Además, conocerás una propiedad especial de Scratch: A saber, puedes iniciar al mismo tiempo varios programas de Scratch. Gracias a esto se simplifican muchos programas en los que algo debe suceder al mismo tiempo.

En el caso de este modelo se trata de los dos motores del ventilador: Uno debe accionar la hélice, el otro debe girar todo el ventilador hacia la izquierda y hacia la derecha, de forma alternada.

### **1** Tarea de construcción

Monta el ventilador como indica el manual de instrucciones.

## Tareas de programación

### 2 Medir el calor

Primero debes escribir un programa que muestre el valor de medición actual del sensor.

Para ello, como en el contador de monedas, necesitas una variable que tu denominas p. ej. «Valor térmico».



En esas variables debe almacenarse el valor del sensor medido y leído una y otra vez por el controlador ft. El valor real de las variables se muestra automáticamente en Scratch en la «plataforma».



Describe primero el algoritmo y después escribe un programa Scratch que almacene una vez por segundo el valor de medición en las variables.

Para ello necesitas la siguiente orden de Scratch:



Guarda el programa con el nombre «Medición de calor».

Inicia ahora tu programa y apunta el valor de las variables que se muestra en la «plataforma».

Calienta después el termistor, sosteniéndolo con firmeza entre el pulgar y el índice. Apunta otra vez el valor de tus variables después de aproximadamente medio minuto.

Ahora intenta enfriar el termistor (soplar, ventilar, sostener contra un metal frío, colocar en el alféizar de una ventana, o similares), y apunta el valor mostrado.

Necesitamos igualmente esos dos valores.

### 3 Control de niveles

Tu ventilador ahora debe regular la velocidad de la hélice en función del calor en la habitación.

El motor de la hélice puede accionarse con ocho niveles de velocidad (1-8).

Divide ahora el área medida en la tarea 1, entre el valor de medición más reducido y el valor de medición más elevado, en ocho áreas de aproximadamente el mismo tamaño:

Ejemplo:

Nivel del ventilador	Valores de medición
1	20 a 25
2	26 a 30

Tu tabla:

Nivel del ventilador	Valores de medición
1	
2	
3	
4	
5	
6	
7	
8	

## Tareas de programación

### 3 Control de niveles - Continuación

La velocidad de la hélice ahora debe regularse de modo que ésta gire más rápido en tanto haga más calor.

Primero describe verbalmente el algoritmo.

Consejo: Para ello debes comparar un valor límite de un área, después de otra, con el valor de medición.

Para la programación en Scratch necesitas una orden de programación que

re



en

da  
valor límite:

compa-  
el valor  
término  
medido  
la entra-  
con un

Escribe un programa Scratch, de manera que la velocidad del motor M1 (motor de la hélice) se adapte al calor en la habitación, y pruébalo.

La velocidad de la hélice debe mostrarse en una variable «Nivel».

Guarda el programa con el nombre «Stufensteuerung Propeller».

Ade-  
ne-  
una  
del

«En  
...



más,  
cesitas  
variante  
ele-  
mento  
caso de  
entonces»:

Si no se cumple con la condición detrás de «en caso de», entonces se ejecuta el comando que está indicado en «de lo contrario».

## Tareas de programación

### 4 Giro del ventilador

El motor con el ventilador está montado en un disco giratorio que es accionado por un motor. Al girar, respectivamente un componente presiona hacia la izquierda y hacia la derecha un botón pulsador para detectar la «posición final».

En esos puntos el motor debe invertir su dirección.

Escribe un programa Scratch de manera que el motor del ventilador gire de forma

continua desde la izquierda hacia la derecha, y pruébalo.

Consejo: Ya has aprendido cómo funciona en la barrera. Simplemente adapta un poco tu programa «barrera», y ya estará listo.

Guarda el programa con el nombre «Giro del ventilador».

### 5 Ventilador inteligente

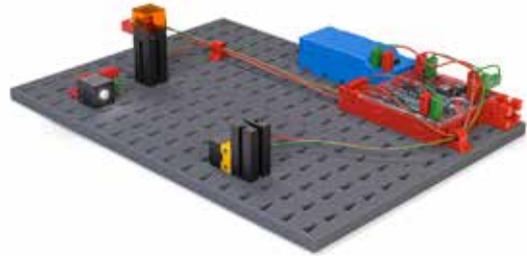
Para que tu programa pueda girar el ventilador al mismo tiempo y adaptar la velocidad de la hélice debes iniciar al mismo tiempo ambos programas.

Para ello añade tu programa de la tarea 3 al programa de la tarea 4 y guárdalo con el nombre «Ventilador inteligente».



## Modelo 7: Telégrafo– Transmitir & codificar señales

¿De qué se trata? Transmitir señales por el aire fue por mucho tiempo un sueño de la humanidad. Cuando eso finalmente fue posible, esa tecnología permitió primero el desarrollo de muchas cosas, p. ej. del teléfono. ¡Conoce tú mismo los primeros pasos de la transmisión de señales!



### Introducción

Uno de los inventos más significativos del siglo XIX es la telegrafía – la transmisión rápida de mensajes a grandes distancias.

Hoy para nosotros es natural que podamos hablar por teléfono con otras personas en todo el mundo, como si estuvieran junto a nosotros. Pero esto no siempre fue así. Durante mucho tiempo, los mensajeros y el correo eran la única posibilidad para transmitir mensajes a grandes distancias.

A veces un mensaje podía tardar algunos días hasta llegar a un destinatario.

Con la energía eléctrica fue posible después una transmisión a la velocidad de la luz.

Con este modelo conocerás un telégrafo simple que funciona con luz.

En este modelo deberás medir el tiempo. En Scratch esto se realiza de forma muy sencilla con las variables ya disponibles «cronómetro».

Esta variable indica los segundos desde el inicio del programa o el último restablecimiento del reloj:



### 1 Indicar el tiempo + restablecer el «cronómetro»

Coloca ahora la marca de verificación en las variables del cronómetro (en el menú «sentir»).

Después el tiempo en segundos se muestra en la «plataforma».

Con la siguiente orden de Scratch puedes restablecer el cronómetro en «0». A continuación, de inmediato se reinicia en segundos.



### 2 Tarea de construcción

Monta el telégrafo como indica el manual de instrucciones: Si se dispone de dos controladores ft, entonces montar por separado el emisor y el receptor. En caso contrario montar el telégrafo («estación Morse») que contiene tanto un emisor, como también un receptor.

## Modelo 7: telégrafo– Transmitir & codificar señales

¿Órdenes de programación que se repiten a menudo? Ahorra tiempo con «bloques»

En los programas en esta tarea propuesta se repiten más a menudo secuencias de órdenes.

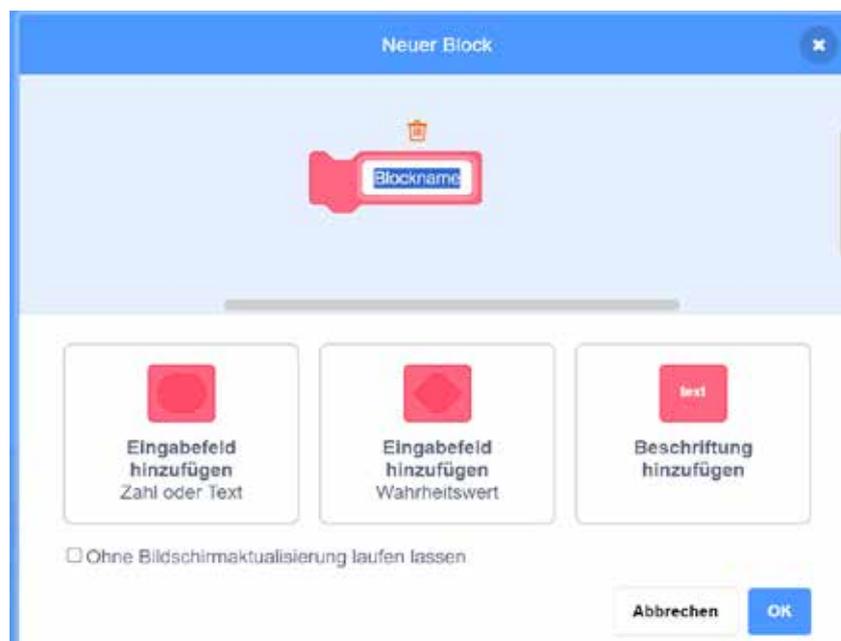
Para hacer más claros tus programas Scratch, Scratch te brinda la posibilidad de formar bloques propios («subprogramas») a partir de las secuencias de órdenes de esa clase, que también después puedes activar como una orden de Scratch.

Para crear un bloque propio de esa clase debes hacer clic en el botón «nuevo bloque»:

### Meine Blöcke

Neuer Block

Y después ingresar un nombre propio para tu subprograma. Ese nombre ahora se encuentra en tu nuevo bloque. Ahora puedes incorporarlo en tus programas, como una orden de Scratch.



## Tareas de programación

### 1 Señal luminosa

El telégrafo funciona de modo muy similar a la barrera de luz que ya has conocido en el sistema de alarma: El LED forma el emisor y el fototransistor el receptor.

El emisor envía un signo luminoso y el receptor detecta que la luz incide en el fototransistor.

Inicia ahora como programa emisor el programa de luz intermitente del primer modelo.

Nota: Si se trabaja con emisor separado y receptor, el emisor, junto con el LED de emisión también debería encender o apagar el LED rojo, para que detecte que se envía una señal. Completa el programa de modo correspondiente. Guarda el programa con el nombre «Emisor».

Programa un programa Scratch para el receptor que espere a que el fototransistor detecte una señal luminosa del emisor.

Después debe encenderse el LED de recepción, y debe apagarse nuevamente tan pronto como el emisor ya no detecte ninguna señal luminosa.

Nota: Si se trabaja con emisor separado y receptor, entonces se prueba el programa del receptor, iniciando el programa «emisor» en el emisor.

Guarda el programa del receptor con el nombre «Receptor».

En la estación Morse ahora debes iniciar al mismo tiempo el programa del «emisor» - (luz intermitente) - y el programa del «receptor». Ya has aprendido cómo funciona en el ventilador.

Guarda el programa de emisión y recepción combinado con el nombre «Emisor - Receptor».

## Tareas de programación

### 2 Transmisión de un número

Hasta el momento tu telégrafo puede enviar solo una señal individual («encender», «apagar»). Esa es muy poca información para un mensaje.

Ahora queremos intentar transmitir más información: El emisor debe transmitir números al receptor. Para ello, el emisor debe parpadear varias veces y el receptor debe contar las señales de «encender» enviadas.

Programa para enviar:  
Reescribe primero el programa de emisión de la tarea anterior de manera que envíe un número predeterminado de señales de «encender».

Para ello puedes utilizar la orden de Scratch "repetir ... veces":



Guárdalo con el nombre «Emisor de número».

Programa para recibir:  
El receptor debe guardar el número de las señales recibidas en una variable «número».  
Ya has aprendido cómo funciona en el modelo «Contador de monedas»: El programa Scratch del receptor, para ello, en cada nueva señal de «encender» debe aumentar en uno la variable «número».

Reescribe tu programa de receptor anterior de manera que cuente las señales del emisor.

Consejo: Después de que hayas aumentado en uno el valor de la variable «número» debes esperar hasta que la señal del emisor sea nuevamente «apagar», para que no cuentes una señal varias veces.

Prueba el programa junto con el programa del emisor y guárdalo con el nombre «Receptor de número».

Nota: En la «estación Morse», como en la tarea anterior, se deben iniciar y guardar juntos el programa de emisión y el de recepción.

## Tareas de programación

### 3 Emisor de código numérico

Con el programa de la tarea 2 ahora puedes transmitir precisamente un número. Pero para un mensaje completo esto todavía no es mucho.

Con un pequeño complemento de tu programa de emisión ahora puedes transmitir una secuencia de números.

Para que el receptor sepa cuándo ya se ha transmitido un número y sigue uno nuevo, el emisor, después de cada número, debe hacer una pausa mayor, de dos segundos.

Completa primero tu programa de emisor de la tarea 2, de manera que transmita varios números de forma consecutiva.

Para que tu programa Scratch sea más claro ahora debes definir un bloque propio «número de emisión» que se encargue de la transmisión de un número.

Funciona de este modo: Haces en Scratch un bloque propio que transmita un número correspondiente de señales de «encender» y después haga una pausa de dos segundos.

Al crear el bloque debes hacer clic en «añadir campo de entrada» y establecer para el bloque el campo de entrada «Número».

En tu programa de emisor ahora puedes activar varias veces el bloque de forma consecutiva, como una orden de Scratch, para enviar distintos números.

Guarda el programa con el nombre «Emisor de código numérico».



## Tareas de programación

### **4** Receptor de código numérico

El programa adecuado para el emisor de código numérico debe contar las señales de «encender» recibidas.

También aquí debes crear un bloque («número de recepción») propio y programarlo. El número recibido debe mostrarse en la «plataforma».

Consejo: Después de cada señal el bloque «número de recepción» debe iniciar el cronómetro y comprobar si dura más de 1,5 segundos hasta que llega la siguiente señal.

Para ello, utiliza el cronómetro.

Amplía tu programa Scratch de la tarea **2** de modo correspondiente.

Pruébalo con el emisor de código numérico de la tarea anterior y guárdalo con el nombre «receptor de código numérico».

Nota:

Si utilizas la «estación Morse», también aquí debes iniciar y guardar juntos ambos programas.

## Tarea experimental

### 5 Texto de mensaje

Con tu telégrafo de código numérico ahora puedes transmitir mensajes completos. Numera para ello las letras del alfabeto («A» = 1, «B» = 2, ...):

A	
B	
C	
D	
E	
F	

G	
H	
I	
J	
K	
L	

M	
N	
O	
P	
Q	
R	

S	
T	
U	
V	
W	
X	

Y	
Z	

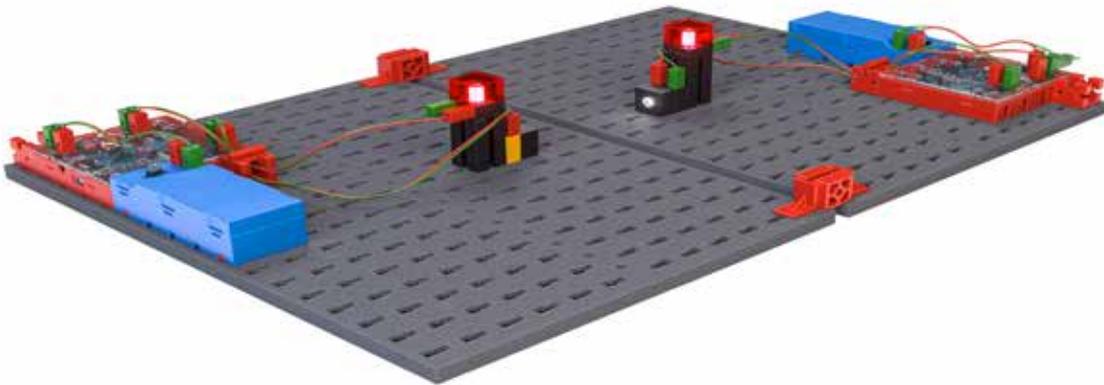
Piensa un mensaje y reemplaza (codifica) las letras mediante el respectivo número en el alfabeto. Envía el mensaje adaptando de modo correspondiente el programa de la tarea 3.

Inicia el programa de receptor de la tarea 4. El receptor debe registrar los números recibidos y «decodificar», por tanto, reemplazar nuevamente por las letras del alfabeto.

Prueba si se puede recibir correctamente un mensaje de texto así de largo.

## Modelo 8: Cifrado

*¿De qué se trata? ¿Finalmente intercambiar mensajes con tu mejor amiga o con tu mejor amigo de forma segura? En este modelo aprenderás mucho sobre la seguridad en la transmisión de mensajes. Y también sobre por qué debe prestarse atención a la seguridad también en la Internet.*



### Introducción

La transmisión de mensajes con señales luminosas (modelo de telégrafo) tiene una desventaja: Si alguien sostiene un fototransistor en el trayecto de transmisión también puede leer el mensaje transmitido.

Un cifrado del mensaje protege frente a

esto.

En este modelo aprenderás sobre dos procedimientos de cifrado sencillos.

Las líneas telefónicas, las redes de wifi y la conexión a Internet también pueden interceptarse y, por eso, en la actualidad, si es posible, se protegen mediante cifrado.

### **1** Tarea de construcción

Para esta tarea necesitas la estación de telégrafo de la tarea propuesta anterior.

# Tareas de programación

## 2 Cifrado de reemplazo

Tu transmisión de mensajes con señales luminosas (en el telégrafo) ahora protegerse de que alguien intercepte la transmisión mediante un cifrado.

Un método de cifrado muy antiguo es la utilización de un alfabeto en el que las letras se cambian unas por otras.

Eso se realiza muy fácilmente, numerando las letras del alfabeto no de 1 a 26, sino asignándoles aquí en la tabla los números de 1 a 26 de forma aleatoria.

denomina «Cifrado de reemplazo» porque caracteres individuales se reemplazan por otros.

Piensa en una numeración aleatoria de las letras y regístrala en la siguiente tabla. Presta atención a utilizar cada uno de los números 1-26 solo una vez:

Codifica ahora el texto «Hola mundo» con ese alfabeto y transmítelo con el programa de tu estación de telégrafo.

A	
B	
C	
D	
E	
F	

G	
H	
I	
J	
K	
L	

M	
N	
O	
P	
Q	
R	

S	
T	
U	
V	
W	
X	

Y	
Z	

Piensa: ¿Cuál es la desventaja de ese cifrado?

Ese procedimiento de cifrado también se

## Tareas de programación

### 3 Cifrado de desplazamiento

El acuerdo de un alfabeto completo como «secreto», que deben conocer el emisor y el receptor del mensaje cifrado, es muy complicado: Siempre debes tener a mano la tabla completa.

Ante todo, eso es muy difícil si deseas intercambiar mensajes cifrados con muchos receptores distintos que solo pueda descifrar el respectivo receptor; se vuelve complicado.

Por eso, ya Julio César en el siglo I a.C. inventó un cifrado para el que es suficiente con una «clave» muy sencilla, que incluso se puede identificar: Él simplemente «desplazó» los números de las letras del alfabeto solo en un número fijo.

Tenía que recordar la "clave" que el emisor y el receptor debían conocer; solo el número de lugares en los que se desplazaban los números.

El receptor simplemente debe restar de nuevo el valor de la clave al descifrarla, y encontrará la letra correcta.

Puedes programar este cifrado muy fácilmente, sumando la clave a cada letra (el número de la tabla) en tu mensaje.

Para que el número no sea mayor que 26 se hace un truco: divides números más grandes por 27 y después calculas el resto.

Aquí encontrarás una ventaja de los ordenadores: aun cuando no puedas o todavía no puedas calcular algo así: ¡Puedes programarlo y el ordenador calculará por ti! :)

El receptor hace lo mismo, solo «hacia atrás»: La clave se resta del número recibido; el resto después de dividir por 26 es el

número (= la letra del alfabeto) del mensaje original.

Las siguientes operaciones de Scratch «+», «-» y «módulo» se encargan de los pasos del cálculo:



Cam-  
tu pro-  
emisión  
ma de  
modelo  
de manera que

bia ahora  
grama de  
y tu progra-  
recepción al  
telégrafo,

1. a la variable «número» (por tanto, el número de la letra que deseas enviar), antes del envío, se suma la variable «clave» módulo 27, y

2. después de la recepción de «número» a la clave se sustraiga módulo 27.

Prueba el programa con el texto «Hola mundo» y guárdalo con el nombre «Emisor con cifrado» o «Receptor con descifrado» (en el modelo estación de telégrafo «emisor-receptor con cifrado y descifrado»).

Piensa: ¿Cuántas claves diferentes puedes seleccionar en ese cifrado?

## Tarea experimental

### 4 Salida de voz

Tu programa de recepción ahora debe emitir las letras del mensaje recibido como voz.

Prueba tu programa y guárdalo con el nombre «Salida de voz receptor con cifrado».

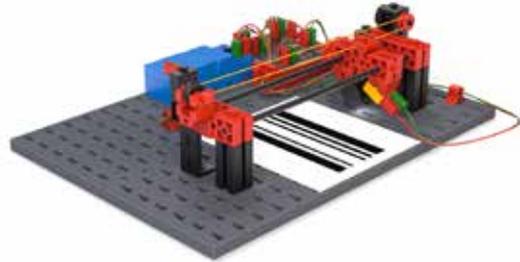
Ya aprendiste cómo funciona la salida de voz en los modelos «barrera» y «contador de monedas».

Programa un bloque «Salida de voz» propio al que puedas transferir el número descifrado, y que reproduzca las letras correspondientes.

Consejo: La salida de voz requiere algo de tiempo. Para que tu receptor mientras tanto no “pase por alto” ninguna señal, debes aumentar la pausa de 2,5 a 3 segundos en el emisor, después de enviar un número.

## Modelo 9: Lector de código de barras

¿De qué se trata? ¿Alguna vez viste ese patrón de barras en los envases? Detrás se oculta un código para el que construirás aquí una «máquina lectora».



### Introducción

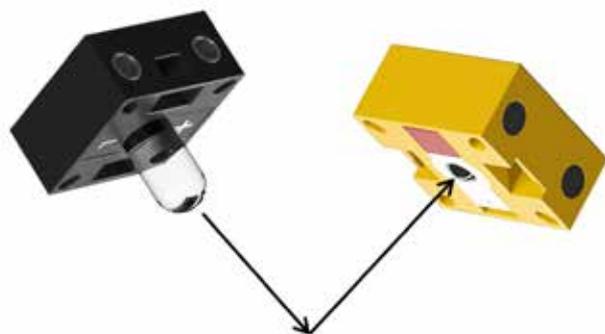
En el telégrafo aprendiste cómo se pueden transmitir mensajes por medio de señales luminosas entre dos microcontroladores (ordenadores). De ese modo, dos ordenadores pueden intercambiar información entre sí, por tanto, pueden «entenderse».

Este modelo se ocupa de un problema similar: ¿Cómo puede «leer» información un microcontrolador (o un ordenador)?

También este problema tiene dos aspectos: En primer lugar (como la «señal» para el intercambio de mensajes) necesitas una representación adecuada de la información, una «codificación». Y también necesitas un mecanismo para «leer» esa codificación.

En este modelo aprenderás cómo se expresa información mediante «barras» - como código de barras (= «bar-code») - y cómo el microcontrolador puede leerla con una clase de «barrera de luz indirecta» (como en el sistema de alarma) que está montado en el lado inferior de la «cabeza de lectura» en el lector de código de barras.

Cómo funciona la «barrera de luz indirecta» es muy fácil de explicar: Un fondo claro refleja el haz de luz del LED, de manera que la barrera de luz se cierra (cambia el fototransistor). En el caso de un fondo oscuro no alcanza suficiente luz el fototransistor y la barrera de luz está interrumpida (el fototransistor no cambia).



### 1 Tarea de construcción

Monta el lector de código de barras como indica el manual de instrucciones. Presta atención a que la «cabeza de lectura» con el fototransistor y el LED se deslicen libremente sobre los ejes y a que los cables conectados sean suficientemente largos para

## Tareas de programación

### 2 Mover la cabeza de lectura

Primero debes dar vida a tu lector de código de barras con un pequeño programa: La cabeza de lectura primero debe desplazarse a la posición inicial, bien a la izquierda, en la posición final izquierda (el botón pulsador se activa) y detenerse allí.

Después de un segundo de tiempo de espera, la cabeza de lectura debe entonces comenzar la «pasada de lectura», moviéndose hacia la derecha (no demasiado rápido), hasta alcanzar la posición final derecha.

Debe detenerse allí.

Escribe un programa Scratch correspondiente, adapta la velocidad del motor y guarda el programa con el nombre «Mo-

### 3 Detectar barras

Ahora tu lector de código de barras debe detectar líneas negras. Amplía tu programa de manera que en la pasada de lectura (de izquierda a derecha), de una variable «línea» indique el valor «1» cuando la cabeza de lectura detecte una línea negra, y el valor «0» cuando no detecte ninguna línea.

Haz que el valor de las variables se muestre en la «plataforma».

Ahora coge una hoja de papel blanca (tamaño: DIN A5). Con un marcador un poco grueso pinta algunas líneas verticales en la mitad superior de la hoja y pásalas por debajo de la cabeza de lectura.

Inicia tu programa y prueba si reconoce todas las líneas.

¿Son lo suficientemente anchas para ser detectadas? ¿La cabeza de lectura es suficientemente lenta para no pasar por alto ninguna línea?

Nota: En el momento en el que tu cabeza de lectura comienza con la lectura detecta en breve una línea, aunque no haya ninguna allí.

Ese error se evita si al iniciar el programa en Scratch, con la siguiente orden, estableces que el sensor esté conectado a «I4»:



## Tareas experimentales

### 5 Leer código numérico

Ahora puedes hacer que tu lector de código de barras detecte un código numérico completo.

Al igual que el telégrafo, el lector de códigos de barras ahora debe aprender a leer varios valores numéricos de forma consecutiva.

Para ello, entre dos valores numéricos («grupos de barras») debes establecer una distancia (p. ej. 1 cm) en la que el lector de código de barras pueda detectar que comienza un nuevo número y que debe poner el contador en «0».

Prueba qué tan apretadas puedes dibujar las barras para que aún puedan contarse correctamente.

Consejo: Si llevas las barras demasiado cerca del borde de la hoja, la cabeza de lectura no las alcanzará y tampoco podrá detectarlas. Marca en tu hoja con un trazo de lápiz los bordes que la cabeza de lectura ya no puede alcanzar.

Mide el tiempo que la cabeza de lectura necesita para moverse en la distancia entre dos «grupos de barras». Adapta tu programa de manera que el lector de código de barras detecte esas distancias y cuente varios grupos de barras de forma consecutiva.

Guarda tu programa con el nombre «Leer

código numérico».

Para recordar: Debes indicar como número decimal los tiempos menores a un segundo (en el modo de escritura en inglés en Scratch con un «.» en lugar de una coma), p. ej. «0,3».

### 6 Emitir código numérico

Ahora tu lector de código de barras también debe decir los números leídos.

Adapta tu programa de modo correspondiente y guárdalo con el nombre «Emitir código numérico».

Consejo: La salida de voz toma algo de tiempo. En ese caso la cabeza de lectura puede pasar por alto una línea. Para que eso no pase deberías detener brevemente el motor y reiniciarlo después de la salida de voz.

## Modelo 10: Robot buggy

¿De qué se trata? En «rojo» detenerse, en «verde» circular ... Un semáforo regula el tráfico para que todas las calles puedan usarse o cruzarse de forma segura. Construyes tu propio semáforo para camino peatonal, que incluso se pone en «verde» al pulsar el botón.



### Introducción

En este modelo construirás un pequeño «robot buggy», un robot conductor de tres ruedas. A continuación, lo controlarás a distancia.

El robot buggy tiene dos motores que respectivamente accionan una rueda más grande, así como una «rueda de apoyo» pequeña.

Para hacer que el robot buggy marche derecho debes accionar ambas ruedas en la misma dirección.

Si el robot buggy debe girar, las ruedas deben accionarse en una dirección opuesta entre sí.

Como «mando a distancia» sirve el teclado de tu ordenador o tableta.

En Scratch, con el siguiente comando Scratch puedes consultar si están presionadas teclas individuales del teclado de



tu ordenador o de tu tableta:

Como teclas para el mando a distancia p. ej. puedes seleccionar letras o las teclas de flechas y la barra espaciadora.

En  
ta-  
vin-  
sí



una sub-  
rea, debes  
cular entre  
condi-

### 1 Tarea de construcción

Monta el robot buggy como indica el manual de instrucciones.

## Tareas de programación

### 2 Mando a distancia 1

Escribe ahora un pequeño programa Scratch que consulte de forma continua las teclas «barra espaciadora», «flecha hacia arriba» y «flecha hacia abajo».

Si se presiona la tecla «flecha hacia arriba», el robot buggy debe desplazarse hacia delante; si se presiona la tecla «flecha hacia abajo», debe desplazarse hacia atrás. Si se presiona la barra espaciadora ambos motores deben detenerse de inmediato.

Consejo: Si los motores no rotan en la dirección deseada debes cambiar la conexión verde y la roja.

Guarda el programa con el nombre «Mando a distancia 1».

### 3 Mando a distancia 2

El mando a distancia ahora debe ampliarse con la posibilidad de girar el robot buggy hacia la izquierda o derecha: Si se presiona la tecla «flecha hacia la izquierda», debe girar en sentido antihorario; si se presiona la tecla «flecha hacia la derecha» debe girar en sentido horario.

¿Qué comandos debes completar para ello en tu programa Scratch?

Observa: Durante el giro los motores deben accionar las ruedas lentamente para que puedas controlar el robot buggy con la mayor precisión posible.

Guarda el programa con el nombre «Mando a distancia 2».

### 4 Mando a distancia 3

Ahora mediante tu mando a distancia también debe poder regularse la velocidad del robot buggy. Para ello, guarda la velocidad (nivel 0-8) que debe seleccionarse en el desplazamiento recto o hacia atrás, en una variable («Velocidad»).

Si se presiona la tecla «+» la velocidad debe aumentarse en un nivel; si se presiona la tecla «-» debe reducirse en un nivel.

Consejo: Solo puedes reducir la velocidad cuando el valor de la variable «velocidad» sea mayor que «0», y solo puedes aumentarla cuando sea menor que «8».

Puesto que el programa se ejecuta muy rápidamente, después de cada cambio de velocidad deberías esperar, por ejemplo, medio segundo («0,5») para que la velocidad no se cambie en varios pasos a la vez.

Observa: La nueva velocidad regulada se regula solo en el robot buggy si presionas la tecla para el desplazamiento recto o el desplazamiento hacia atrás.

Amplía tu programa «mando a distancia 2» de modo correspondiente y guárdalo con el nombre «Mando a distancia 3».

## Tareas experimentales

### 5 Parada de emergencia

Scratch también puede acceder al micrófono de tu ordenador, de tu tableta, de tu teléfono inteligente o de tu cámara web, y determinar el volumen sonoro en la habitación. Puedes hacer que el volumen sonoro medido actualmente con el micrófono se muestre en la «plataforma» colocando una marca de verificación junto a la variable «volumen sonoro»

en el  área «sentir».

Ahora tu pequeño robot buggy no solo se detendrá de inmediato cuando presiones la barra espaciadora, sino también aplaudiendo fuerte.

Primero prueba qué valor muestra la variable «volumen sonoro» cuando aplaudes. Amplía tu programa «mando a distancia 3» en esa «parada de emergencia»: Si el volumen sonoro es mayor que un valor que has definido, entonces ambos motores deben detenerse de inmediato.

### 6 Salida de voz

Ya conociste la salida de voz en la barrera.

Amplía ahora tu programa de mando a distancia de manera que comandos que indicas al robot buggy mediante el teclado («hacia delante», «hacia atrás», «hacia la izquierda», «hacia la derecha», «parada», «más rápido», «más lento»), se reproduzcan con sonido como comando.

Guarda el programa con el nombre «Mando a distancia con salida de voz».

## Modelo 11: Robot pintor

*¿De qué se trata? ¿Una máquina pintora? Puedes hacer que ese robot pintor pinte modelos automáticamente.*



### Introducción

*En esta tarea propuesta el robot buggy se convierte en un robot pintor.*

*Podrá pintar de modo independiente imágenes simples, desde un triángulo, pasando por un cuadrado, hasta una «casa sin levantar el lápiz». Para lograr esto, debes medir el tiempo que el robot pintor necesita para un movimiento determinado.*

*Esto se realiza fácilmente con el «cronómetro» de Scratch que has conocido en la tarea propuesta para el telégrafo.*

Stoppuhr

setze Stoppuhr zurück

### **1** Tarea de construcción

*Reforma el robot buggy de la tarea propuesta 9 anterior en el robot pintor, como indica el manual de instrucciones.*

## Tareas de programación

### 2 Control del robot pintor

En el modelo del telégrafo aprendiste los subprogramas. Con ello, los programas se vuelven más claros y cortos.

Carga tu programa «mando a distancia 3» de la tarea propuesta del robot buggy y reemplaza con subprogramas las secciones del programa para desplazamiento recto y desplazamiento hacia atrás, girar y detenerse.

Con ese programa ahora puedes hacer que el robot pintor dibuje.

Pruébalo y guarda tu programa como «Control del robot pintor».

### 3 Medición de tiempo

Ahora, con el control del robot pintor de la tarea 1, dibuja algunas formas geométricas simples: un triángulo, un rectángulo y un cuadrado.

Para que el robot pintor después pueda «seguir pintando» por sí solo el dibujo, debes medir el tiempo que necesita para desplazarse recto o girar en un ángulo.

Agrega, para ello, una variable «Duración» en tu control, en la que después de una «parada» se registre el tiempo del cronómetro.

Observa: El cronómetro muestra la cantidad de segundos con un «valor decimal» (un número decimal). En Scratch en lugar de una coma se utiliza un punto

Amplía el programa de control de modo correspondiente y apunta los tiempos que necesita el robot buggy para girar y desplazarse de forma recta al pintar un triángulo y un cuadrado.

Guarda el programa como «Medición robot pintor».

### 4 Robot pintor

Ahora puedes hacer que el robot pintor pinte.

Cambia el programa de la tarea 2 de manera que no se mida el tiempo, sino que los movimientos para dibujar un triángulo (o un cuadrado) se realicen de forma consecutiva, por un tiempo predeterminado.

Prueba el programa y adapta las indicaciones de tiempo de manera que el robot pintor dibuje un triángulo (o cuadrado) lo más preciso posible.

Guarda el programa con el nombre «triángulo» (o «cuadrado»).

## Tarea experimental

### 5 Casa sin levantar el lápiz

Ahora haz que el robot pintor dibuje en un trazo la «Casa sin levantar el lápiz».

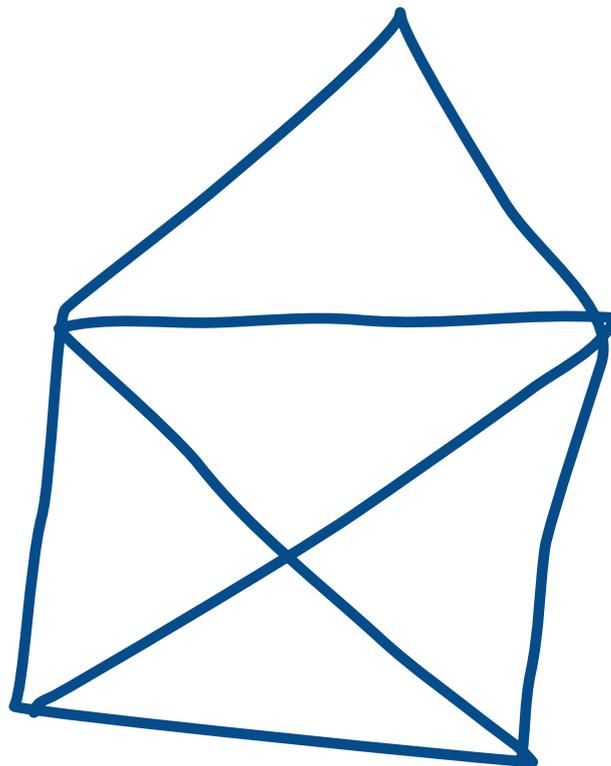
En este caso, necesitas distintos ángulos para que pueda girar, y diferentes longitudes laterales.

Primero intenta dibujar la «casa sin levantar el lápiz» con el control de la tarea 1..

Con el programa de la tarea 2, mide los tiempos para los distintos ángulos y secciones.

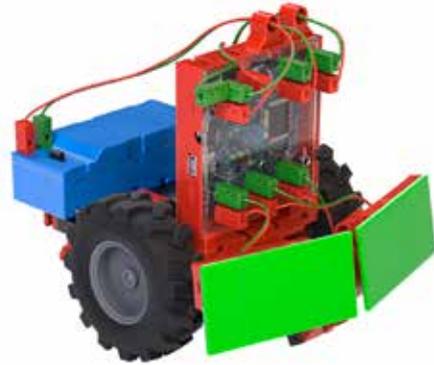
Reescribe ahora el programa de la tarea 3, de manera que el robot pintor pinte la «casa sin levantar el lápiz» en un trazo y de forma independiente.

Guarda el programa como «Casa sin levantar el lápiz».



## Modelo 12: Mayordomo

¿De qué se trata? Un robot pintor está muy bien, pero este robot incluso puede encontrar su camino, prestando atención a esquivar obstáculos. Después de esto ya casi puedes programar autos sin conductor.



### Introducción

En las tareas para este modelo conviertes el robot buggy en un vehículo autónomo que puede transportar objetos: un pequeño «mayordomo».

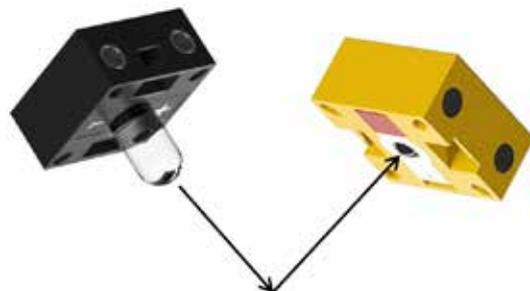
Para poder detectar y esquivar obstáculos, el mayordomo, adelante a la izquierda y a la derecha, tiene en cada caso un botón pulsador que se activa cuando el mayordomo entra en contacto con un objeto.

Para que el mayordomo también detecte líneas de límite negras en el lado inferior está provisto de un fototransistor y de un LED - un tipo de «barrera de luz indirecta»:

Un fondo claro refleja el haz de luz, de manera que la barrera de luz se cierra.

En el caso de un fondo oscuro no alcanza suficiente luz el fototransistor y la barrera de luz se interrumpe.

Ya conoces el principio del lector de código de barras.



### 1 Tarea de construcción

Monta el mayordomo como indica el manual de instrucciones.

Si ya has construido el robot buggy o el robot pintor puedes pasar directamente al paso 15 en el manual de instrucciones.

## Tareas de programación

### 2 Poner la mira en el objetivo

El mayordomo, después de que fue orientado hacia un objetivo, debe desplazarse hacia allí de forma recta hasta que se detiene debido a un aplauso fuerte (véase la tarea propuesta para el robot buggy). Intenta seleccionar la velocidad de los dos motores de manera que el mayordomo se desplace lo más recto posible.

Escribe un programa Scratch que controle el mayordomo de modo correspondiente, pruébalo y guárdalo con el nombre «Poner la mira en el objetivo». Utiliza para ello tus subprogramas del robot pintor.

### 3 Avisador sonoro

Tan pronto como el mayordomo haya alcanzado su objetivo y se haya detenido debe emitir una señal sonora.

Consejo: En el sistema de alarma has aprendido cómo puedes reproducir un sonido.

Completa tu programa Scratch con un avisador sonoro y guárdalo con el nombre

«Alcanzar el objetivo».

### 4 Detección de líneas

Puesto que se accionan ambas ruedas de distintos motores, el mayordomo no se desplaza exactamente de forma recta. Ante todo, cuando las dos ruedas se desplazan sobre un terreno diferente, el mayordomo puede desviarse de su curso. Para que el mayordomo no abandone un área predeterminada ahora debe aprender a detectar una línea de limitación. Utiliza para ello el recorrido que se adjunta con el kit.

Si el mayordomo detecta con su «barra de luz» que ha alcanzado la línea de limitación debe retroceder un poco para apartarse de la línea, girando, y después continuar con el desplazamiento.

Completa tu programa Scratch de la tarea 2 de modo correspondiente. Pruébalo y guárdalo con el nombre «Detección de líneas».

## Tarea experimental

### 5 Esquivar obstáculos

Ahora puede suceder que el mayordomo se choque contra un obstáculo en el trayecto hacia el objetivo.

Para que el mayordomo no quede atascado allí, cuando los sensores frontales detectan el obstáculo (por tanto, se presiona el botón pulsador izquierdo o derecho), debe retroceder un poco y, después, esquivar el obstáculo.

Piensa cómo puedes programar del modo más simple para que se esquiven los obstáculos.

Amplía tu programa para detectar y esquivar obstáculos.

Prueba el programa y guárdalo con el nombre «Esquivar obstáculos».

