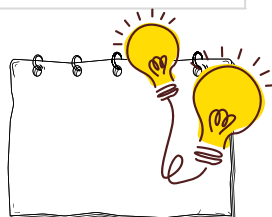


Nivel principiante		
RETO	(✓)/X	COMENTARIO
Reto 1. Semáforo peatonal simplificado		

Nivel intermedio		
RETO	(✓)/X	COMENTARIO
Reto 2. Contador de segundos.		
Reto3. Luz de emergencias.		
Reto 4. Personaliza colores.		

Nivel avanzado		
RETO	(✓)/X	COMENTARIO
Reto5. Semáforo inteligente con sensor.		
Reto 6. Dos semáforos sincronizados.		

Mis notas



"Programa financiado por el Ministerio de Educación, Formación Profesional y Deportes"



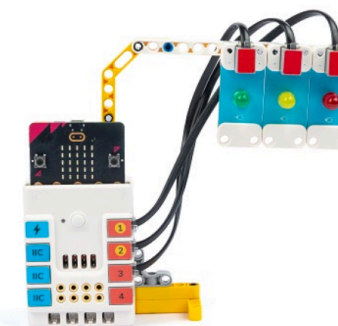
MINISTERIO DE EDUCACIÓN, FORMACIÓN PROFESIONAL Y DEPORTES



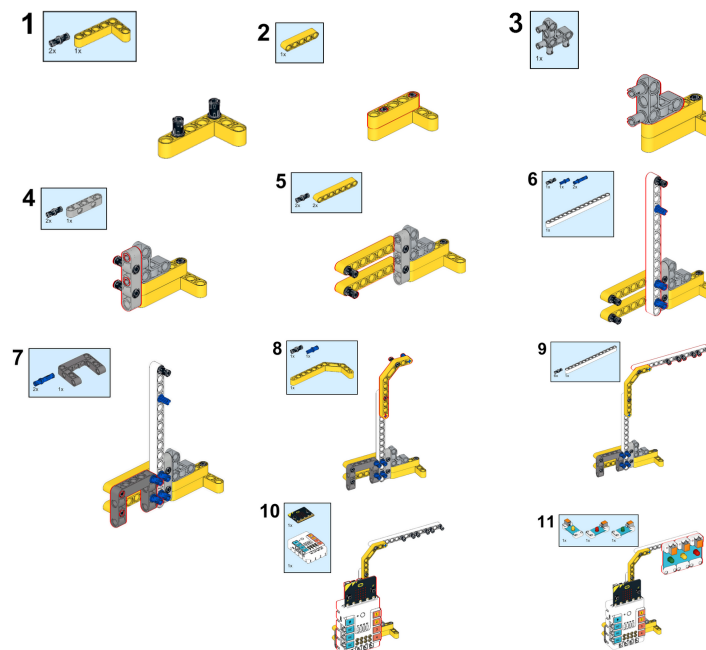
PRIMARIA - CE4.0_M © 5/2/2025 by Código Escuela 4.0_M is licensed under [CC BY-NC-SA 4.0](https://creativecommons.org/licenses/by-nc-sa/4.0/)

EL SEMÁFORO

- En este proyecto, los alumnos construirán un modelo de semáforo utilizando el Kit Nezha V2 y la placa micro:bit.
- Aprenderán a encender y apagar luces LED en diferentes colores (rojo, amarillo, verde) y a programar secuencias de cambio de luces, simulando el funcionamiento real de un semáforo para vehículos o peatones.
- El proyecto fomenta el aprendizaje de la programación por bloques (MakeCode), el uso de hardware, y desarrolla competencias como la lógica, la resolución de problemas y la creatividad. Además, permite relacionar la tecnología con situaciones reales de la vida cotidiana, como la seguridad vial.



1ª PARTE: CONSTRUCCIÓN

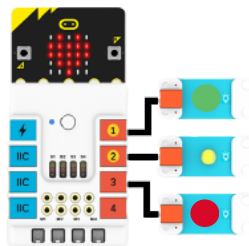


2º Parte. Programación



Programa el semáforo que has construido las lucepara que las se enciendan así:

● Verde → 5 segundos → ● Amarillo → 2 segundos → ● Rojo → 5 segundos → vuelve a empezar.



Funcionamiento

Paso 1. Luz verde encendida

J1 se enciende (verde).
J3 se apaga (rojo).
Espera 5 segundos.
Simula la luz verde del semáforo (los coches pueden pasar).

Paso 2. Luz amarilla encendida

Apaga J1 (verde).
Enciende J2 (amarillo).
Espera 2 segundos.
✓ Simula la luz amarilla del semáforo (precaución).

Paso 3. Luz roja encendida

Apaga J2 (amarillo).
Enciende J3 (rojo).
Espera 5 segundos.
✓ Simula la luz roja del semáforo (los coches se detienen).



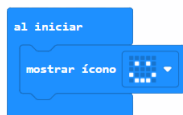
Después de los 5 segundos vuelve a empezar

LED

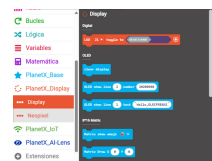
Un LED es una pequeña luz que funciona con electricidad.
LED significa en inglés Light Emitting Diode, que quiere decir Diodo Emisor de Luz.

- Sirve para iluminar cosas: lámparas, televisores, semáforos, luces de coches... ¡hasta lucecitas en los juguetes!
- Gasta muy poca electricidad, por eso se usa mucho.
- Puede tener muchos colores: rojo, verde, azul, amarillo...
- Es pequeño y no se calienta mucho.

1 Empezamos por algo que ya conocemos... bloques de entrada.



2 Descargar la extensión PlanetX



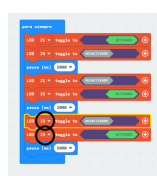
3 Paso 1, Luz verde encendida



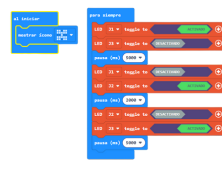
4 Paso 2, Luz amarilla encendida



5 Paso 3, Luz roja encendida



6 Programa final



Retos por niveles

● Nivel 1. Principiante

Reto 1 – Semáforo peatonal simplificado

Añadir un botón: si se presiona, encender rojo inmediatamente (peatones cruzan).

Tras 5 seg, volver a secuencia normal.

Objetivo: introducir interacción física (entrada) y relacionar tecnología con seguridad vial.

● Nivel 2. Intermedio

Reto 2 – Contador de segundos

Mientras está la luz verde: mostrar en la pantalla LED de micro:bit la cuenta regresiva (5, 4, 3, 2, 1). Igual para el rojo.

Objetivo: manejar variables y usar bucles y mostrar datos en pantalla.

Reto 3 – Luz de emergencia

Añadir condición: si se presiona el botón A → parpadean todas las luces rápidamente (emulando emergencia). Al soltar el botón, vuelve la secuencia normal.

Objetivo: comprender el uso de condicionales y eventos simultáneos.

Reto 4 – Personaliza colores

Diseñar colores diferentes (p. ej. semáforo festivo). Crear nuevas secuencias.

Objetivo: desarrollar creatividad y explorar funciones del kit Nezhá.

● Nivel 3. Avanzado

Reto 5 – Semáforo inteligente con sensor

Integrar un sensor ultrasónico: si detecta "coches" (mano u objeto) a menos de 10 cm → pasa a verde antes de lo habitual. Si no, ciclo normal.

Objetivo: introducir sensores externos.

Comprender condicionales basadas en datos físicos.

Reto 6 – Dos semáforos sincronizados

Crear dos semáforos: uno para coches y otro para peatones.

Programar:

- Cuando luz verde coches → rojo peatones.
- Cuando rojo coches → verde peatones.

Objetivo: lógica más compleja y coordinar procesos paralelos.

Nivel principiante		
RETO	(✓)/X	COMENTARIO
Reto 1. Programa que la puerta abra cuando alguien esté entre 5 y 15 cm y se mantenga cerrada si no hay nadie.		

Nivel intermedio		
RETO	(✓)/X	COMENTARIO
Reto 2. Además de abrir/cerrar la puerta: Mostrar en el micro:bit la distancia en cm. Añadir un tiempo de apertura de 3 segundos.		

Nivel avanzado		
RETO	(✓)/X	COMENTARIO
Reto3. Construir una puerta inteligente		

Mis notas

```

para siempre
si Ultrasonic sensor 31 > distance: cm > 2 y Ultrasonic sensor 31 < distance: cm < 20 entonces
  mostrar icono
  Set 360° servo S1 angle to 120
  pausa (ms) 5000
si no
  mostrar icono
  Set 360° servo S1 angle to 180

```



"Programa financiado por el Ministerio de Educación, Formación Profesional y Deportes"



MINISTERIO DE EDUCACIÓN, FORMACIÓN PROFESIONAL Y DEPORTES



PRIMARIA - CE4.0_M © 5/2/2025 by Código Escuela 4.0_M is licensed under [CC BY-NC-SA 4.0](https://creativecommons.org/licenses/by-nc-sa/4.0/)

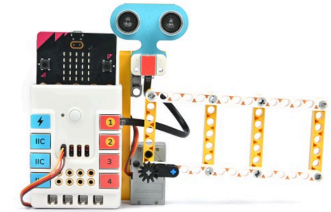
Puerta automática

En este proyecto se construirá una compuerta ultrasónica con el Kit de Invención Nezha V2.

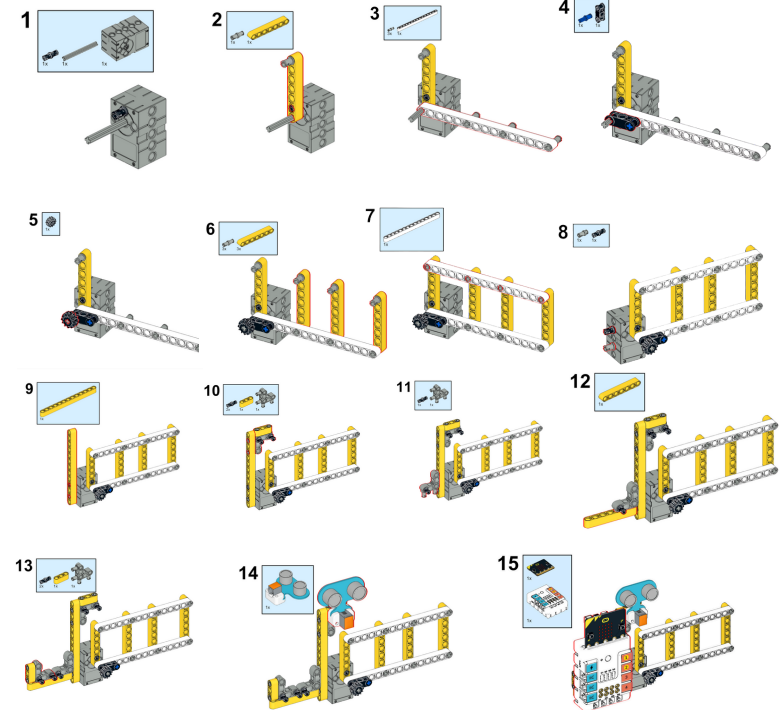
Se aprenderá cómo funcionan los sensores ultrasónicos y su programación.

Se explorará cómo controlar puertas automáticamente según la distancia detectada.

Se discutirán ideas sobre programación, rangos de movimiento y diseños personalizados.

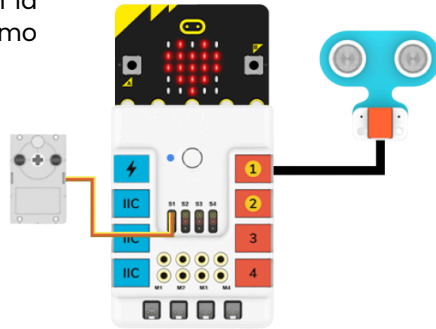


1ª PARTE: CONSTRUCCIÓN



2º Parte. Programación

Conecte el servo de 360° a S1 y el sensor de sonido ultrasónico a J1 en la placa de expansión Nezha como muestra la imagen.



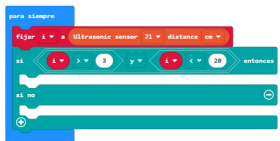
1 Es la instrucción para mover un servomotor usando la placa Nezha.



2 El sensor ultrasónico conectado al puerto J1 de la placa Nezha (PlanetX). Devuelve la distancia en centímetros. Esa distancia se guarda en la variable i. Ejemplo: si hay un objeto a 10 cm, i será 10.



3 Aquí se pregunta: ¿La distancia i es mayor que 3 cm y menor que 20 cm? Si sí, significa que hay un objeto detectado entre 3 y 20 cm de distancia.



4 Si la condición se cumple: Manda al servo 360° conectado en S1 la orden de moverse a una velocidad. 240 indica más velocidad en un sentido. En servos 360°, valores mayores de 180 hacen girar el servo más rápido hacia un lado. Básicamente hace girar la compuerta para abrirla. Espera 5 segundos (5000 ms) antes de seguir con el código. → Mantiene la compuerta abierta durante ese tiempo.

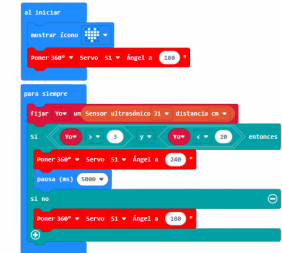


5 Si la condición NO se cumple (es decir, no hay objeto entre 3 y 20 cm), entonces se ejecuta. Envía al servo la posición 180: En servos 360°, 180 suele ser velocidad neutra o girar más lento, dependiendo del modelo. Es probable que aquí signifique cerrar la compuerta o detenerla.



Funcionamiento

Lee la distancia de un objeto con el sensor ultrasónico.
Si algo está entre 3 y 20 cm:
Abre la compuerta (gira el servo a velocidad 240).
Espera 5 segundos.
Si no hay nada en ese rango:
Mantiene la compuerta cerrada o parada (servo a 180).
Repite esto sin parar.



Retos por niveles

Nivel 1 — Principiante

Objetivo: programa que la puerta abra cuando alguien esté entre 5 y 15 cm y se mantenga cerrada si no hay nadie.

Indicaciones:

Usa el sensor ultrasónico para leer la distancia.

Si la distancia está entre 5 y 15:

Servo gira (p.ej. set angle 240).

Si no: servo se detiene (p.ej. set angle 180).

Nivel 2 — Intermedio

Objetivo: además de abrir/cerrar la puerta:

Mostrar en el micro:bit la distancia en cm.

Añadir un tiempo de apertura de 3 segundos.

Indicaciones:

Usa basic.showNumber para mostrar distancia.

Cuando detectes a alguien:

Mueve el servo.

Pausa 3 segundos.

Cierra la puerta.

Nivel 3 — Avanzado

Objetivo: construir una puerta inteligente.

Diferenciar entre "persona cerca" y "objeto pequeño".

Si la distancia está entre 5 y 15 cm Y permanece estable más de 2 lecturas consecutivas → abrir puerta.

Si la distancia varía mucho (e.g. objeto pasa rápidamente), no abrir.

Mostrar un icono especial si detecta movimiento inestable.

Indicaciones:

Guarda varias lecturas en un array o variables.

Compara si la distancia es estable.

Usa iconos (check ✓, x ✗) para indicar puerta abierta/cerrada.

Ejemplo de lógica:

Guardar las 3 últimas lecturas.

Si las 3 lecturas están dentro de ±2 cm → abrir.

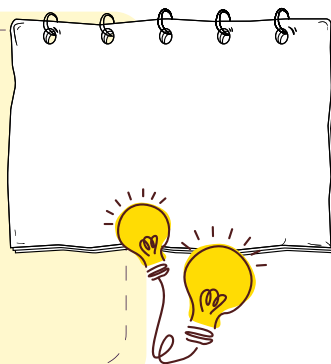
Si no → no abrir.

Nivel principiante		
RETO	(✓)/X	COMENTARIO
Reto 1. Mostrar diferentes iconos según la humedad		

Nivel intermedio		
RETO	(✓)/X	COMENTARIO
Reto 2. Contador de tierra seca		

Nivel avanzado		
RETO	(✓)/X	COMENTARIO
Reto3 . Programa luces intermitentes (parpadeo) cuando la tierra esté muy seca (menos de 20). Además, guarda el valor más bajo de humedad que se haya medido y muéstralo con el botón B		

Mis notas



Medimos la humedad

Este programa empieza mostrando un corazón en la pantalla.

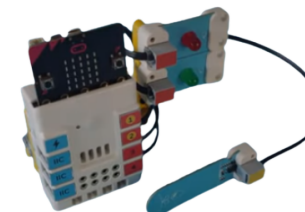
Mide si la tierra está seca o húmeda con un sensor.

Si la tierra está seca (menos de 30), enciende una luz LED en un pin.

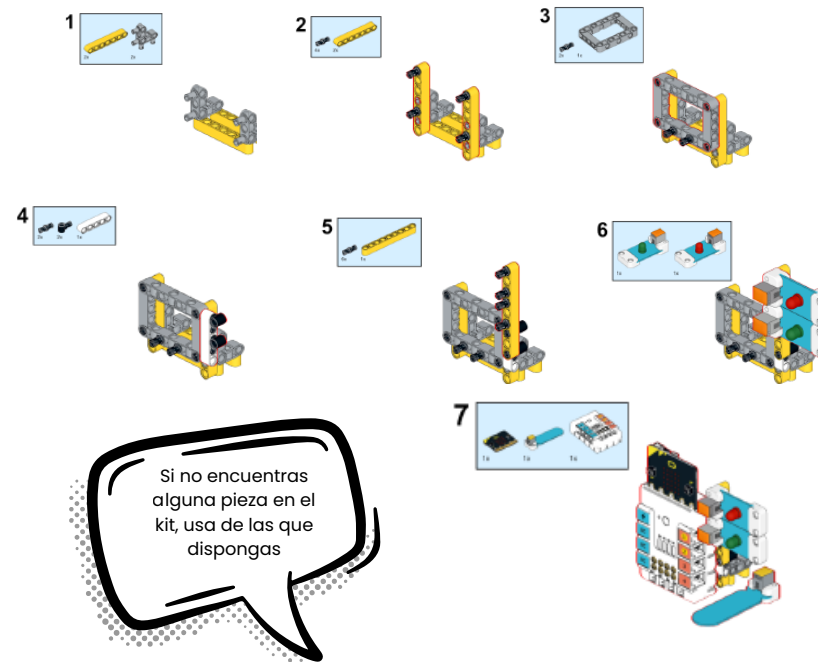
Y apaga otra luz LED en otro pin.

Si la tierra está húmeda, hace lo contrario.

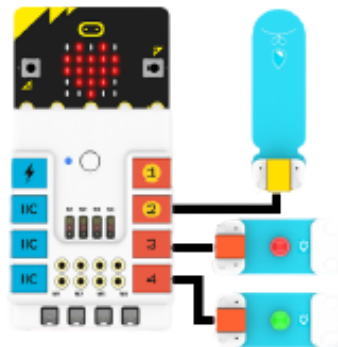
Así sabrás si hay que regar tus plantas.



1ª PARTE: CONSTRUCCIÓN



2º Parte. Programación



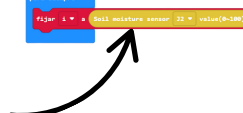
1

Seleccionar imagen a mostrar al comenzar

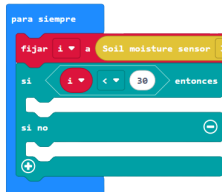


2

Crear variable "i"
Fijar variable "i" al sensor de humedad



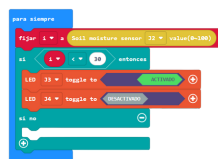
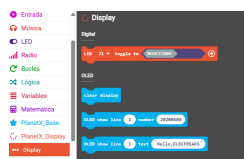
3



Condional
"variable" = "humedad"
es inferior a 30
entonces...

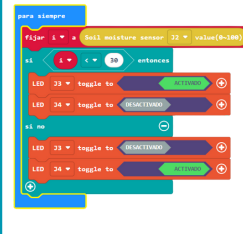
4

¿Dónde encuentro el bloque?



5

Programa final :
cuando la humedad
es menor que 30 led
rojo se enciende J3 y
J4 se apaga si no ...
J4 encendido y J3
apagado



Funcionamiento

Primero se ve un corazón en el micro:bit.
Se mide la humedad de la tierra en el pin J2.
Si la tierra está seca (humedad baja), se enciende la luz del pin J3.
Y se apaga la luz del pin J4.
Si la tierra está húmeda, se apaga J3 y se enciende J4.
Sirve para avisar cuándo la planta necesita agua.



Sensor de humedad

El sensor de humedad funciona midiendo la conductividad eléctrica de la tierra.
Cuando la tierra está húmeda, contiene más agua y permite que la electricidad fluya mejor entre los dos electrodos del sensor.
Si la tierra está seca, la conductividad baja porque hay menos agua para conducir la electricidad.
El sensor convierte esa conductividad en un valor eléctrico que el micro:bit puede leer como un número.
Con ese número, el programa decide si la tierra está seca o húmeda para avisar si hay que regar la planta.

Retos por niveles

Nivel 1. Principiante

Reto 1: Haz que el micro:bit muestre diferentes iconos según la humedad:

Un corazón si está húmedo.

Una calavera si está seco.

Objetivo: Usar condicionales con diferentes imágenes.

Nivel 2. Intermedio

Reto 2: Añade un contador para saber cuántas veces la tierra estuvo seca.

Muestra ese número en la pantalla al pulsar el botón A.

Objetivo: Usar variables y botones.

Nivel 3. Avanzado

Reto 3: Programa luces intermitentes (parpadeo) cuando la tierra esté muy seca (menos de 20).

Además, guarda el valor más bajo de humedad que se haya medido y muéstralo con el botón B.

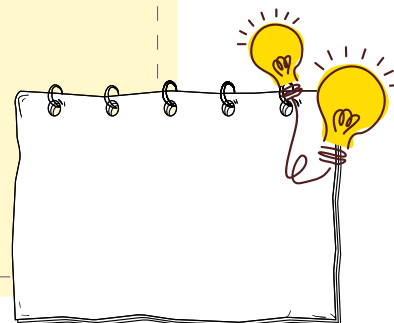
Objetivo: Combinar lógica avanzada, parpadeo de LEDs y memoria de valores mínimos.

Nivel principiante		
RETO	(✓)/X	COMENTARIO
Reto 1. Icono según la línea		

Nivel intermedio		
RETO	(✓)/X	COMENTARIO
Reto 2. Sonidos		

Nivel avanzado		
RETO	(✓)/X	COMENTARIO
Reto 3. Velocidad		

Mis notas



"Programa financiado por el Ministerio de Educación,
Formación Profesional y Deportes"



MINISTERIO
DE EDUCACIÓN, FORMACIÓN PROFESIONAL
Y DEPORTES

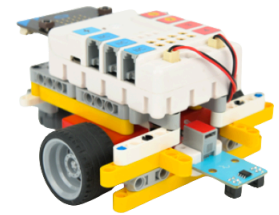


PRIMARIA - CE4.0_M © 5/2/2025 by Código Escuela 4.0_M is licensed under [CC BY-NC-SA 4.0](https://creativecommons.org/licenses/by-nc-sa/4.0/)

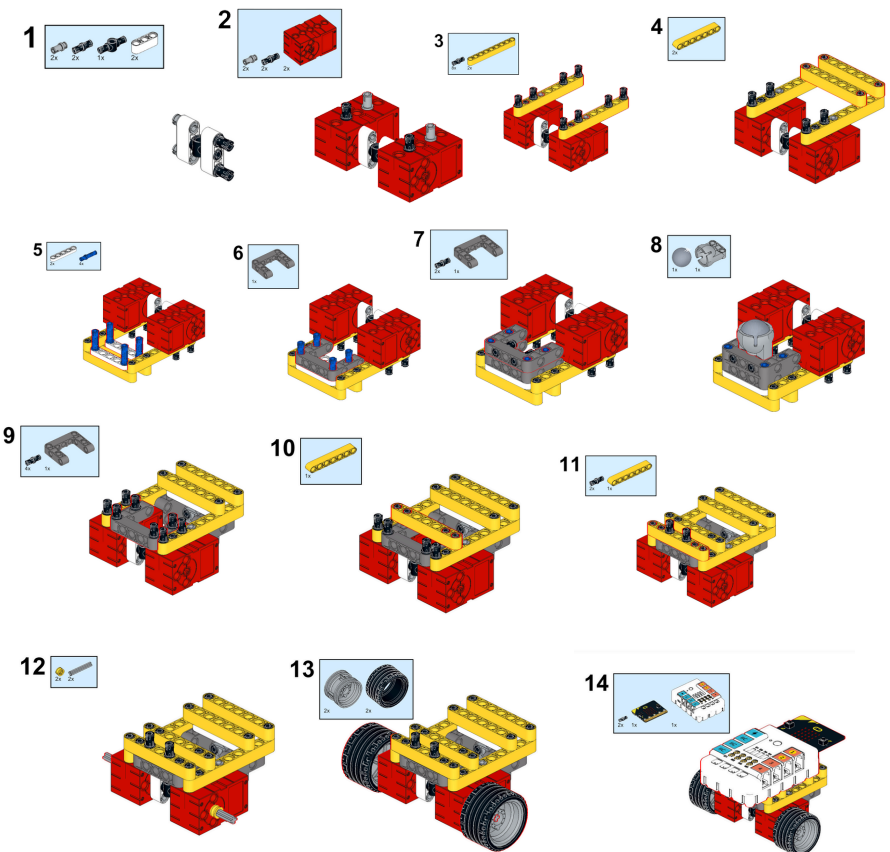
Coche seguidor de líneas

¿Qué hace este proyecto?

Muestra un corazón en la pantalla al comenzar.
Usa un sensor seguidor de línea conectado al puerto J1.
Si detecta línea a la izquierda → gira hacia la izquierda.
Si detecta línea a la derecha → gira hacia la derecha.
Si está centrado sobre la línea → avanza recto

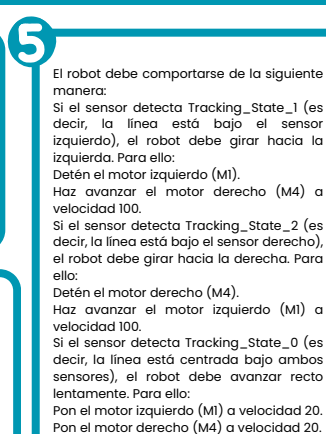


1ª PARTE: CONSTRUCCIÓN





2º Parte. Programación



Actividad: Programa un robot NeZha con micro:bit para que siga una línea negra sobre fondo blanco utilizando el sensor sigue líneas PlanetX conectado al puerto RJ 45.



Retos por niveles

Si está centrado → mostrar un corazón.

Si sigue recto → que no suene nada o reproduzca un tono suave.

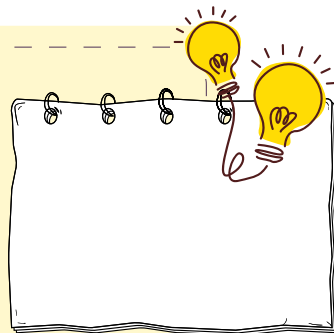
Usar variables para guardar el tiempo o los estados anteriores.

Nivel principiante		
RETO	(✓)/X	COMENTARIO
Reto 1. Detener y mostrar icono		

Nivel intermedio		
RETO	(✓)/X	COMENTARIO
Reto 2. Giro simple para evitar obstáculo		

Nivel avanzado		
RETO	(✓)/X	COMENTARIO
Reto 3. Ajuste de velocidad según distancia.		

Mis notas

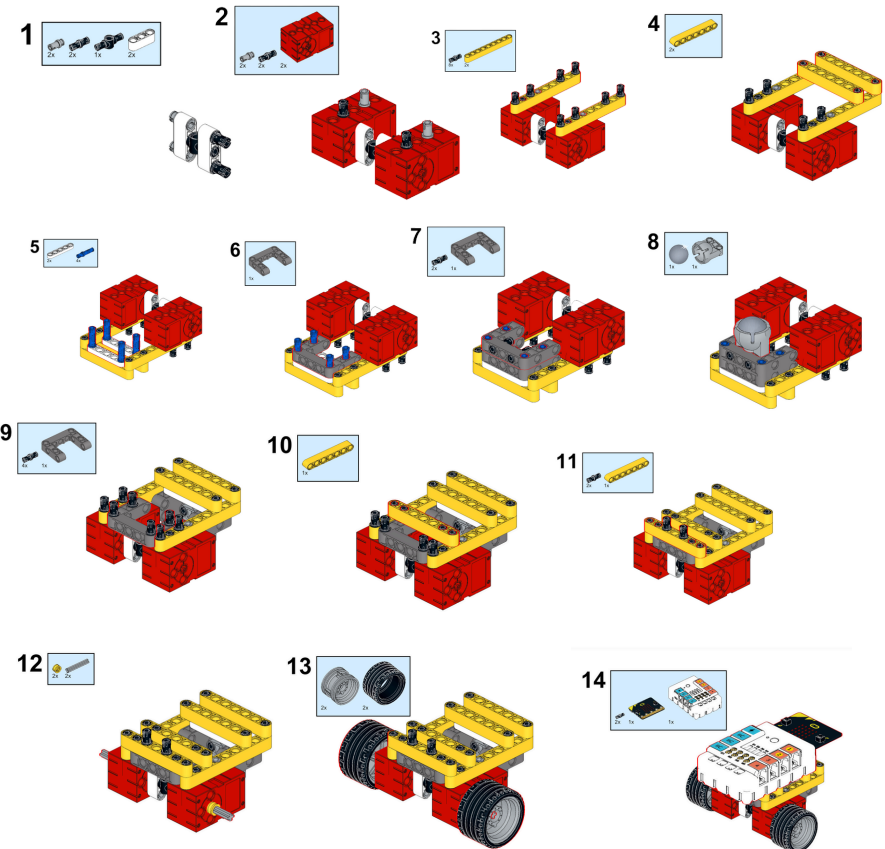


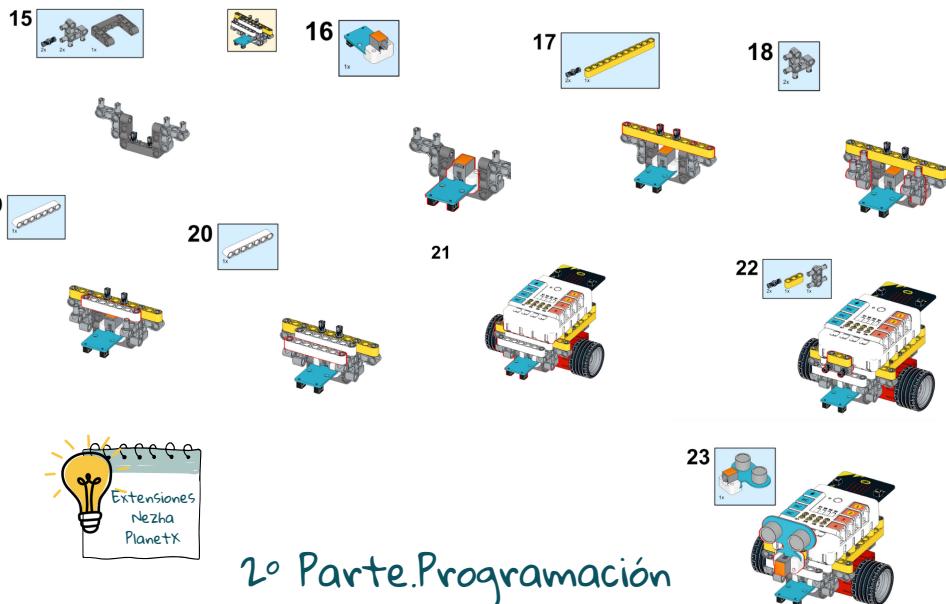
Evación de obstáculos

En este proyecto utilizaremos el Kit de Invención Nezha V2 para construir un coche inteligente capaz de seguir una línea negra y esquivar obstáculos. Aprenderemos cómo funcionan los sensores de seguimiento de línea y los sensores ultrasónicos, y veremos cómo aplicarlos en la creación de nuestro vehículo.



1ª PARTE: CONSTRUCCIÓN





1

si iniciar

mostrar Icono

Crea la variable i

para siempre

fijar i = a

2

Se declara la variable i para almacenar la distancia que mide el sensor ultrasónico.

para siempre

fijar i = a

si y entonces

si no

Condición: Si la distancia está entre 2 y 20 cm, el robot detiene todos los motores durante 2 segundos (pausa), para evitar chocar con algo cercano.

3

para siempre

fijar i = a Ultrasonic sensor 32 distance cm

si i > 2 y i < 20 entonces

stop all motor

pausa (ms) 2000

si no

4

El robot está diseñado para seguir una línea, pero si detecta un obstáculo cerca (entre 2 y 20 cm), se detiene por 2 segundos para evitar colisiones. Luego sigue su camino normalmente usando el sensor de seguimiento de línea.

5

para siempre

fijar i = a Ultrasonic sensor 32 distance cm

si i > 2 y i < 20 entonces

pausa (ms) 2000

si no

Line-tracking sensor 31 si > entonces

Set motor M1 speed to 100

Set motor M2 speed to 100

Line-tracking sensor 31 si < entonces

Set motor M1 speed to 30

Set motor M2 speed to 30

Line-tracking sensor 31 si > entonces

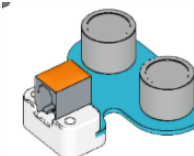
Set motor M1 speed to 100

Set motor M2 speed to 100

Funcionamiento

Sensor Ultrasónico

Un sensor ultrasónico es un dispositivo que mide la distancia a un objeto usando ondas sonoras de alta frecuencia (ultrasónicas). El sensor emite un sonido que rebota en el objeto y regresa al sensor; midiendo el tiempo que tarda ese eco en volver, calcula qué tan lejos está el objeto. Se usa para evitar choques o detectar obstáculos en robots y otros dispositivos.



Retos por niveles

Nivel 1. Principiante

Reto 1: Detener y mostrar icono

Cuando el sensor ultrasónico detecte un objeto entre 2 y 20 cm, que el robot se detenga y muestre una carita feliz 😊 en la pantalla durante 2 segundos, antes de seguir moviéndose.

Nivel 2. Intermedio

Reto 2: Giro simple para evitar obstáculo

Modifica el programa para que, cuando detecte un obstáculo con el sensor ultrasónico (entre 2 y 20 cm), el robot se detenga, gire hacia la derecha durante 1 segundo y luego continúe siguiendo la línea.

Nivel 3. Avanzado

Reto 3: Ajuste de velocidad según distancia

Haz que el robot ajuste la velocidad de los motores según la distancia que mida el sensor ultrasónico:

Si la distancia es menor a 10 cm, los motores funcionen al 30% de velocidad.

Si la distancia es entre 10 y 20 cm, funcionen al 60%.

Si no hay obstáculo cerca (más de 20 cm), funcionen al 100%.

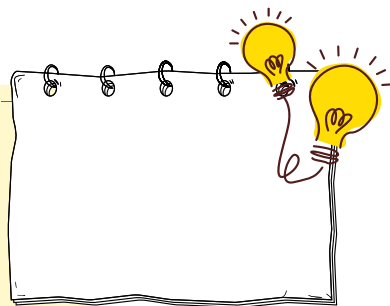
Esto hará que el robot se mueva más despacio cuando haya algo cerca para evitar choques.

Nivel principiante		
RETO	(✓)/X	COMENTARIO
Reto 1. Reto 1: Icono según velocidad		

Nivel intermedio		
RETO	(✓)/X	COMENTARIO
Reto 2. Dos motores controlados		

Nivel avanzado		
RETO	(✓)/X	COMENTARIO
Reto 3: Velocidad y sentido de giro		

Mis notas



Programa financiado por el Ministerio de Educación, Formación Profesional y Deportes



MINISTERIO DE EDUCACIÓN, FORMACIÓN PROFESIONAL Y DEPORTES



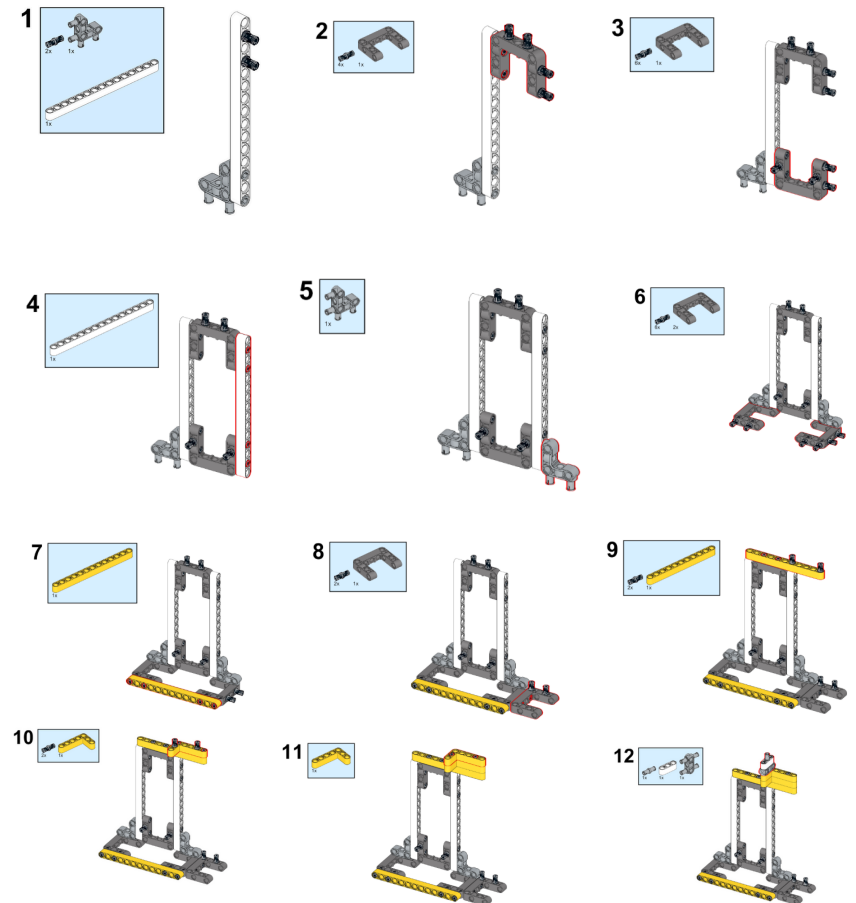
PRIMARIA - CE4.0_M © 5/2/2025 by Código Escuela 4.0_M is licensed under [CC BY-NC-SA 4.0](https://creativecommons.org/licenses/by-nc-sa/4.0/)

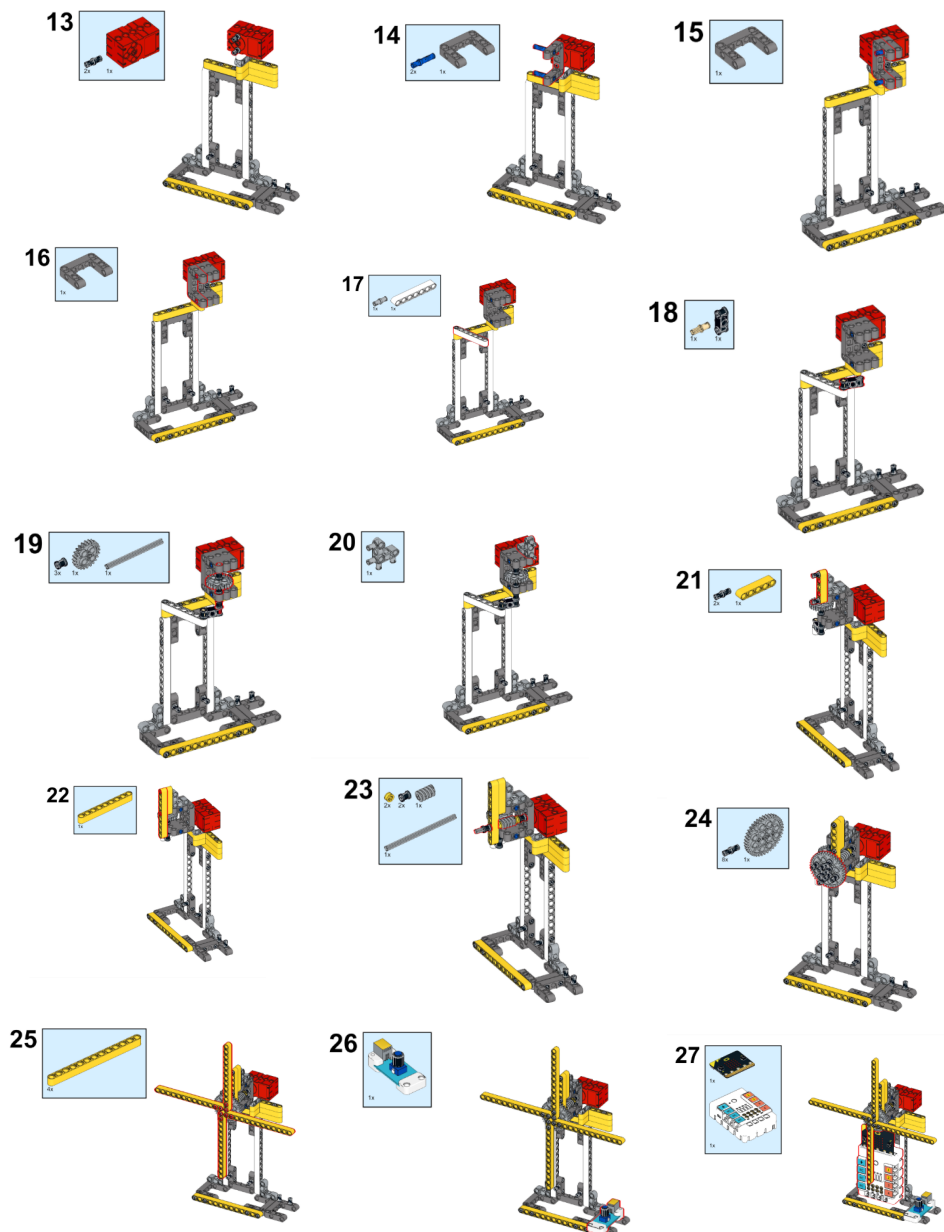
Ventilador oscilante

El ventilador oscilante es un aparato muy usado que reparte el aire en un área más amplia al mover su cabezal de un lado a otro. Vamos a aprender cómo construir un ventilador oscilante que pueda girar y balancear su cabezal automáticamente, utilizando el Kit de Inventor Nezha V2.



1ª PARTE: CONSTRUCCIÓN





"Programa financiado por el Ministerio de Educación,
Formación Profesional y Deportes"



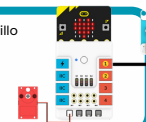
GOBIERNO DE ESPAÑA
MINISTERIO DE EDUCACIÓN, FORMACIÓN PROFESIONAL
Y DEPORTES



PRIMARIA - CE4.0_M © 5/2/2025 by Código Escuela 4.0_M is licensed under [CC BY-NC-SA 4.0](https://creativecommons.org/licenses/by-nc-sa/4.0/)

Potenciómetro

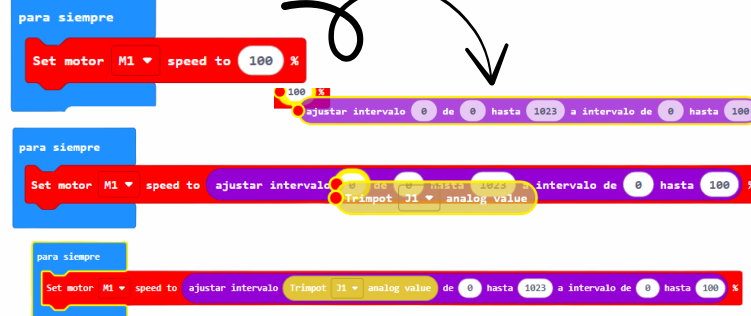
Serve para controlar cosas como la velocidad de un motor, el brillo de una luz o el volumen de un sonido.
Cuanto más lo giras, más sube o baja ese valor.



Es como un "control de volumen", pero para decirle al robot qué tan rápido o fuerte debe hacer algo.

Funcionamiento

2º Parte. Programación



Retos por niveles

Nivel 1. Principiante

Reto 1: Icono según velocidad

Haz que el robot muestre en la pantalla un icono diferente según la posición del potenciómetro:

Si está en menos de 30 → muestra una carita triste 😞

Entre 30 y 70 → muestra una cara neutra 😐

Más de 70 → muestra una carita feliz 😊

Nivel 2. Intermedio

Reto 2: Dos motores controlados

Modifica el programa para que dos motores (M1 y M2) cambien de velocidad dependiendo del potenciómetro:

Si el potenciómetro está por debajo de 512 → solo gira el motor M1.

Si está por encima de 512 → solo gira el motor M2.

Nivel 3. Avanzado

Reto 3: Velocidad y sentido de giro

Crea un programa que use el potenciómetro para controlar la velocidad y la dirección del motor M1:

Si el valor está entre 0 y 512 → el motor gire hacia adelante a velocidad proporcional.

Si está entre 512 y 1023 → el motor gire hacia atrás a velocidad proporcional.

Ejemplo:

Potenciómetro en 256 → motor adelante al 50%.

Potenciómetro en 768 → motor atrás al 50%.

[illegible]