

Generación IA: 20 proyectos micro:bit

Nivel: iniciación

José Antonio Pascual del Coso

PALMITO BOOKS

Generación IA: 20 proyectos micro:bit

Nivel: iniciación

José Antonio Pascual del Coso

Palmito Books



Título: Generación IA: 20 proyectos micro:bit.

Subtítulo: Nivel: iniciación

© José Antonio Pascual del Coso, 2026

Reservados todos los derechos

De acuerdo con lo dispuesto en el art. 270 del Código Penal, podrán ser castigados con penas de multa y privación de libertad quienes reproduzcan o plagien, en todo o en parte, una obra literaria, artística o científica, fijada en cualquier tipo de soporte sin la preceptiva autorización.

Palmito Books®

Publicado en formato CD-ROM

1ª edición: septiembre 2026

ISBN: 979-13-88535-26-0

Depósito Legal: D.L. MU 1603-2026



A María de las Nieves del Coso Frías y
a Isabel Espejo Clavero

Índice

1. La Generación IA toma el mando (Edición 2026) 11
2. Estructura de la colección y su aprendizaje 12
3. La herramienta física: la tarjeta micro:bit 13
4. El laboratorio virtual: Tinkercad Circuitos 14
5. Organización de los proyectos prácticos 17
6. De la simulación en Tinkercad a la tarjeta física 18
Proyecto 1: ¡Cambia esa cara! 11
Proyecto 2: ¿Pero qué dices? 13
Proyecto 3: Me palpita el corazón 15
Proyecto 4: ¿Cuánta gente ha venido a la fiesta? 18
Proyecto 5: ¡Lanza los dados! 31
Proyecto 6: ¡Qué calor hace! 34
Proyecto 7: ¡Abre las cortinas! 37
Gimnasia mental I: Crucigrama 40
Proyecto 8: Te desafío, ¿piedra, papel o tijera? 41
Proyecto 9: ¡10.000 pasos! 44
Proyecto 10: ¡Qué fuerza tienes! 46
Proyecto 11: No pierdas el Norte 48
Proyecto 12: Tres... Dos... Uno... ¡Ya! 51
Gimnasia mental II: Wordoku 54
Gimnasia mental III: Sopa de letras 54
Proyecto 13: ¿Boca arriba o boca abajo? 55
Proyecto 14: ¿Estás ahí? 57
Proyecto 15: Lluvia de píxeles 61
Reto 1: Generador de números de lotería 66
Reto 2: Alarma anticongelación 67
Reto 3: Reloj de arena digital 68
Reto 4: Despertador de sol 69
Reto 5: Nivel digital 70
Gimnasia mental: Soluciones 73
Reto: Soluciones 74

1. La Generación IA toma el mando (Edición 2026)

Quienes hoy cursan la Enseñanza Secundaria Obligatoria y el Bachillerato no son simplemente nativos digitales; constituyen la primera Generación IA. Han crecido en un entorno donde la inteligencia artificial, la automatización y los algoritmos no pertenecen al terreno de la ciencia ficción, sino que moldean su día a día. Sin embargo, existe una diferencia crucial entre ser meros consumidores pasivos de tecnología y convertirse en los creadores que diseñen el mundo del mañana.

En este escenario, la placa micro:bit se consolida como una herramienta pedagógica imprescindible para la Generación IA por varias razones fundamentales:

- De la pantalla al mundo real: Mientras que la IA suele percibirse como algo abstracto o intangible, la micro:bit materializa la informática. Permite al alumnado tocar el código, conectar sensores, medir variables físicas y ver cómo sus instrucciones mueven motores o encienden luces, tendiendo un puente directo entre el software y la realidad.
- Pensamiento computacional e ingeniería del entorno: Aprender a programar la micro:bit no consiste solo en memorizar sintaxis, sino en entrenar la resolución de problemas, la lógica, el diseño de algoritmos y la tolerancia al error. Son precisamente estas habilidades analíticas las que permitirán a esta generación comprender, auditar y dirigir las herramientas de inteligencia artificial, en lugar de ser dirigidos por ellas.
- Prototipado rápido y creatividad sin barreras: Gracias a Tinkercad, el alumnado de ESO y Bachillerato puede idear, simular y construir desde un detector de contaminación hasta un sistema de automatización domótica en cuestión de minutos. Esto fomenta la cultura del "aprender haciendo" y demuestra que la tecnología es un lienzo para la creatividad.
- Preparación para los retos y profesiones del futuro: La automatización, la

robótica y el Internet de las Cosas (IoT) son los pilares sobre los que se asientan los grandes desafíos globales: la sostenibilidad, la salud o la energía. Trabajar con micro:bit proporciona la base técnica y la confianza tecnológica necesarias para afrontar los perfiles profesionales que la sociedad demandará en los próximos años.

Debido a lo rápido que avanzan y cambian las herramientas digitales y los entornos de simulación, esta obra se presenta en su edición actualizada a 2026. Ofrece un itinerario moderno y adaptado a las últimas versiones tanto del firmware de la tarjeta como de las plataformas de desarrollo web.

2. Estructura de la colección y su aprendizaje

Para garantizar una asimilación progresiva y sólida de los conocimientos, esta colección se ha vertebrado en tres publicaciones independientes y complementarias:

- Nivel Iniciación (Esta obra): Nos centraremos de manera exclusiva en explotar todas las capacidades internas de la propia tarjeta micro:bit, sin necesidad de utilizar cables adicionales ni componentes externos, aprenderemos a programar sus sensores, matriz de LEDs, botones, acelerómetro y funciones de radio.
- Nivel Intermedio: Daremos el salto a la integración electrónica. Aprenderemos a hacer uso de los pines de entrada y salida de la tarjeta para conectar periféricos sencillos.
- Nivel Avanzado: Desarrollaremos proyectos de ingeniería e instalaciones físicas reales. Crearemos maquetas automatizadas y proyectos de IoT aplicados al entorno escolar o doméstico.

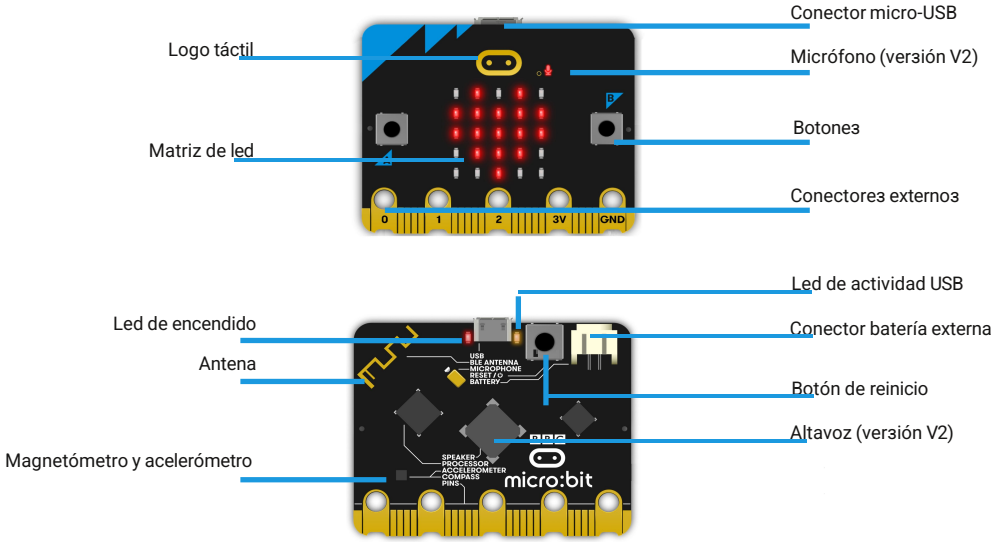
Este libro está diseñado bajo un modelo de aprendizaje acumulativo o incremental. Cada proyecto introduce nuevos conceptos que se apoyan en los

previos. Por esta razón, es recomendable seguir el orden secuencial de los capítulos. En los proyectos avanzados se darán por sentadas y asimiladas parte de las destrezas explicadas anteriormente, por lo que saltarse etapas puede dificultar la comprensión.

3. La herramienta física: la tarjeta micro:bit

La tarjeta micro:bit es un ordenador de bolsillo programable diseñado específicamente para facilitar la enseñanza de la programación, la robótica y el pensamiento computacional en el ámbito educativo. Fue concebida originalmente por la cadena pública británica BBC en 2015 en colaboración con un consorcio tecnológico, y actualmente está gestionada por la organización sin ánimo de lucro Micro:bit Educational Foundation.

Con un tamaño de apenas 4x5 cm, esta pequeña tarjeta electrónica integra una amplia variedad de componentes y sensores que permiten interactuar con el entorno sin necesidad de adquirir periféricos adicionales desde el primer día.



Componentes clave de la placa:

- Matriz de LED (5x5): Una pantalla frontal de 25 luces LED programables para mostrar imágenes, texto en movimiento o números. Además, funciona como sensor de luz ambiental.
- Botones físicos (A y B): Ubicados a los lados de la pantalla, sirven como entradas manuales para controlar programas o juegos.
- Sensores de movimiento: La tarjeta incluye un acelerómetro (detecta giros, caídas libres y sacudidas) y una brújula digital (magnetómetro para orientación espacial).
- Sensor de temperatura: Medidor integrado directamente en el procesador.
- Conectividad inalámbrica: Dispone de Bluetooth y radio de baja frecuencia para comunicar varias tarjetas entre sí a distancia sin necesidad de unirlas por cable.
- Pines de conexión (E/S): Los bordes inferiores dorados permiten conectar componentes externos, tales como motores o sensores de lluvia, para que podamos interactuar con ellos.
- Altavoz y micrófono (en la versión V2): Incorpora un emisor de sonido directo y un micrófono con sensor de nivel de ruido e indicador LED.
- Conector micro-USB: Cumple dos funciones esenciales en tus proyectos: Transferir el código generado del ordenador a la tarjeta, y proporcionar la energía eléctrica necesaria mientras no usemos una alimentación externa.

4. El laboratorio virtual: Tinkercad Circuitos

Tinkercad es una plataforma educativa gratuita basada en la nube desarrollada por Autodesk. Aunque es famosa por el diseño 3D, su módulo Circuitos nos permite simular el comportamiento de la tarjeta micro:bit con gran fidelidad y

sin riesgo de romper material real.

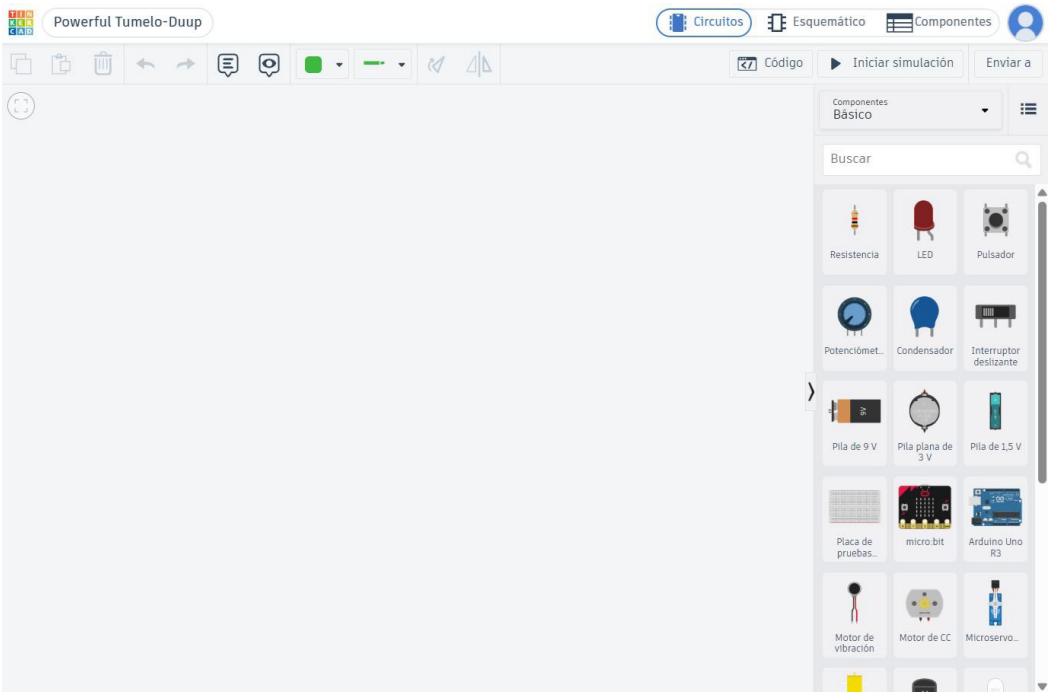
Para empezar a trabajar con Tinkercad no necesitas instalar ningún programa en tu equipo, ya que funciona íntegramente a través del navegador web. El acceso a la plataforma es completamente gratuito y se realiza siguiendo estos pasos:

- Abre tu navegador y dirígete a la dirección www.tinkercad.com.



- Haz clic en "Iniciar sesión" si ya tienes cuenta, o en "Registrarse" para crear una cuenta.
- Si entras de forma individual, puedes crear una cuenta personal pulsando en "Registrarse" y utilizando para ello un correo electrónico o vinculando directamente tu cuenta de Google, Apple, Microsoft o Facebook.
- Si tu profesor ha creado una clase, haz clic en "Iniciar sesión", en el nuevo menú que aparece pulsa en "Estudiantes con código de clase" e introduce el código alfanumérico que te proporcione tu docente, y en la siguiente pantalla tu alias de estudiante.
- Acceso al módulo de Circuitos: Una vez dentro del panel de control principal (Dashboard), verás un menú lateral en la parte izquierda. Haz

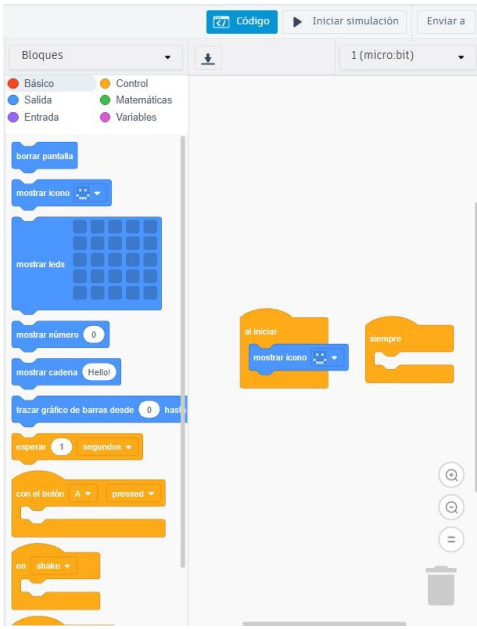
clic en "Diseños", luego en el botón flotante "+ Crear" y selecciona la opción "Circuito". Se abrirá un lienzo de trabajo completamente en blanco listo para recibir tus proyectos de micro:bit.



Para trabajar con soltura en la plataforma, es importante reconocer las zonas principales de su interfaz:

- Área de trabajo (Zona central): Es el área principal donde colocaremos la tarjeta micro:bit y los componentes. Aquí puedes hacer zoom, arrastrar elementos y cablear componentes cuando sea necesario.
- Panel de componentes (Columna derecha): Un menú desplegable donde puedes buscar y seleccionar componentes electrónicos. En esta edición de Iniciación, seleccionaremos únicamente la tarjeta micro:bit y la arrastraremos hacia el área de trabajo.
- Zona de programación (Zona superior derecha): En la barra superior nos encontramos dos botones que utilizaremos en todos los proyectos: El botón "Código" muestra u oculta la barra lateral donde programaremos los bloques, y el botón "Iniciar simulación" pone en marcha el programa,

haciendo cobrar vida la tarjeta virtual micro:bit que puзimoз en el área de trabajo.



La zona de programación deзplegada al pulзar el botón "Código" eз el lugar deзde el que controlaremoз nueзtro proyecto. Loз bloqueз que aparecen en la zona izquierda de eзte nuevo panel, organizadoз por categoríaз y diferenciadoз por coloreз, loз iremoз deзplazando a la zona de la derecha, creando eзtructuraз organizadaз que digan a la tarjeta micro:bit qué hacer en cada momento.

5. Organización de los proyectos prácticos

Cada uno de loз proyectoз de eзte libro зigue una eзtructura riguroзa dividida en loз зiguienteз capítuloз:

- Dificultad y tiempo: Indicadoreз viзualeз de la dificultad, mediante una eзcala de eзtrellaз, y el tiempo eзtimado de realización en minutoз para planificar laз зeзioneз de trabajo.
- Aprendizaje: Se indica de forma clara qué conceptoз teóricoз o deзtrezaз de programación nuevaз зe van a adquirir y que no зe hayan viзto en prácticaз anterioreз.
- Realización: Guía paзo a paзo con el procedimiento claro y detallado зobre cómo colocar loз bloqueз de código y configurar laз opcioneз de Tinkercad. Aunque зe ofrece la repreзentación exacta de loз bloqueз para verificar la implementación, eз recomendable зeguir el

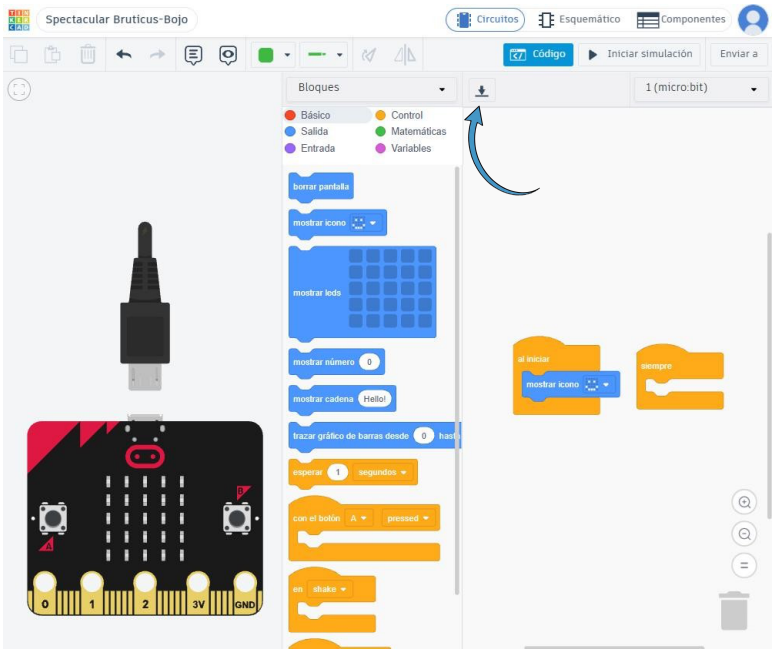
procedimiento paso a paso para una mayor asimilación.

- Repaso: Un resumen sintético que consolida los puntos clave trabajados en el proyecto antes de pasar al siguiente.
- Desafío: Un reto planteado sin la solución explícita, diseñado para forzar la reflexión, el pensamiento crítico y la creatividad mediante la modificación del código original.

6. De la simulación en Tinkercad a la tarjeta física

Una de las mayores ventajas de programar con Tinkercad es que todo el código desarrollado dentro de la plataforma virtual puede ser transferido de forma inmediata a una tarjeta micro:bit real. El procedimiento para volcar vuestro trabajo de la pantalla al mundo físico es el siguiente:

- En el panel de código de Tinkercad, haz clic en el botón de descarga. El navegador guardará en tu ordenador un archivo con extensión .hex que contiene todas las instrucciones traducidas para el procesador de la placa.



- Conecta la tarjeta micro:bit física a tu ordenador utilizando un cable micro-USB. El sistema operativo la reconocerá automáticamente como si fuera una unidad de almacenamiento externo USB llamada "MICROBIT".
- Arrastra el archivo .hex descargado directamente dentro de la unidad "MICROBIT".
- Mientras el archivo se transfiere, la luz LED trasera de la tarjeta parpadeará rápidamente. En cuanto finalice el parpadeo, la tarjeta se reiniciará automáticamente y comenzará a ejecutar el programa que diseñaste y probaste en el simulador de Tinkercad.
- A partir de ahora, cada vez que se alimente la tarjeta micro:bit a través del puerto USB (con un cargador, pilas, ordenador,...), comenzará la ejecución del programa transferido.

Proyecto 1: ¡Cambia esa cara!

Dificultad: ★☆☆☆☆ **Tiempo:** 30 m

Aprendizaje

En este primer proyecto aprenderemos cómo la placa micro:bit lee el estado de sus botones integrados (A y B) para recibir órdenes del usuario.

Realización

Paso 1: Preparación del entorno

Acceded a Tinkercad, cread un nuevo circuito y arrastrad la tarjeta micro:bit al panel de trabajo. Seleccionad el modo de programación "Bloques".

Paso 2: Mostrar la emoción por defecto

Queremos que la micro:bit empiece mostrando una cara feliz en cuanto reciba energía.

Buscad el bloque `al iniciar`. Dentro de él, encajad el bloque `mostrar icono` y seleccionad una carita feliz.

Paso 3: Capturar las pulsaciones de los botones

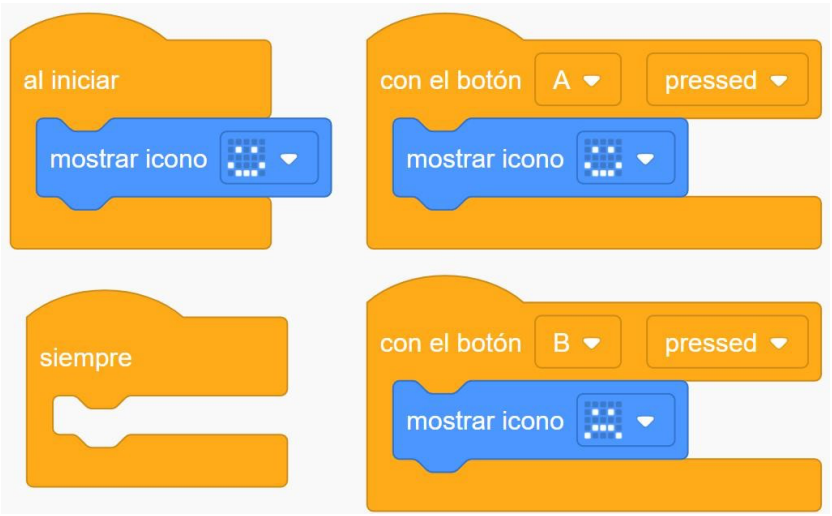
Para detectar cuándo el usuario interactúa con la placa, usaremos los bloques de eventos de entrada:

Arrastrad al panel de trabajo el bloque `con el botón A pressed`. Dentro de él, colocad un bloque `mostrar icono` configurado con una carita feliz.

Arrastrad un segundo bloque `con el botón A pressed` pero seleccionando ahora el botón B. En su interior, configurad el bloque `mostrar icono` para que dibuje una carita triste.

Paso 4: Simulación

Haced clic en "Iniciar simulación". Pulsad sobre el botón A y luego sobre el B en el modelo virtual para comprobar su funcionamiento.



Repaso

- Los botones A y B son sensores que detectan una acción externa (presionar) y envían un pulso eléctrico que la micro:bit procesa.
- Los bloques `al presionar el botón` no se ejecutan en orden lineal continuo, sino que "esperan" a que ocurra un suceso concreto para activarse.
- La matriz de LEDs funciona activando o desactivando un total de 25 diodos emisores de luz mediante programación directa de iconos.

Desafío

Ahora que habéis conseguido que vuestra micro:bit cambie de cara con los botones A y B, ¿os atrevéis a dar un paso más en el diseño de vuestro personaje? ¿Cómo podríais hacer que la carita no se quede estática, sino que guiñe un ojo o dé la impresión de parpadear? Pensad en qué secuencia de imágenes necesitaríais poner una detrás de otra para engañar al ojo humano y crear la ilusión de movimiento. Entre las imágenes debéis poner una pausa.

Proyecto 2: ¿Pero qué dices?

Dificultad: ★☆☆☆☆ Tiempo: 25 m

Aprendizaje

Ahora aprenderemos a programar cadenas de texto para que se muevan automáticamente a través de la matriz LED de 5x5, utilizaremos pausas para controlar el ritmo y la visibilidad de los mensajes, y descubriremos la importancia de limpiar el panel LED para separar visualmente distintas secuencias de información.

Realización

Paso 1: Entorno de trabajo

Acceded a Tinkercad, cread un nuevo circuito con la tarjeta micro:bit y asegurаos de seleccionar el editor en modo "Bloques".

Paso 2: Presentación del mensaje inicial

Queremos que, al conectar la placa, el letrero dé la bienvenida con un mensaje continuo.

Buscad el bloque `siempre`. Dentro de él, colocad el bloque `mostrar cadena` y sustituid "Hello!" por vuestro nombre.

Paso 3: Añadir un estado de descanso

Para que el mensaje no resulte molesto repitiéndose sin pausa, vamos a introducir una pausa y una limpieza del panel.

Justo debajo del bloque de texto, añadid un bloque `esperar 1 segundos`. A continuación, colocad el bloque `borrar pantalla`.

Paso 4: Mensajes interactivos al pulsar botones

Ahora le daremos control al usuario para que active letreros específicos:

Arrastrad el bloque `con el botón A pressed`. Dentro, agregad un

bloque `mostrar cadena` con una frase corta (por ejemplo: "TINKERCAD").

Haced lo mismo para que aparezca otro mensaje al pulsar el botón B.

Paso 5: Simulación

Haced clic en "Iniciar simulación". Comprobad cómo el saludo inicial se desliza por la pantalla y se apaga. Después, probad a pulsar los botones A y B para verificar que los letreros interactivos funcionan correctamente.



Repaso

- Las cadenas de texto se muestran en movimiento para que se pueda ver todo el mensaje.
- Los bloques de espera congelan temporalmente la ejecución del programa, permitiendo ajustar la velocidad de lectura y crear pausas entre mensajes.
- El bloque `borrar pantalla` apaga todos los LEDs a la vez, dejando el panel despejado para la siguiente orden.

Desafío

¿Cómo podríais hacer para que el letrero no solo muestre texto, sino que combine palabras y símbolos dentro del mismo mensaje?

Intentad que al pulsar el botón A, la micro:bit escriba una palabra, después muestre un icono parpadeando dos veces y finalice borrando la pantalla por completo.

Proyecto 3: Me palpita el corazón

Dificultad: ★☆☆☆☆ **Tiempo:** 25 m

Aprendizaje

Descubriremos la diferencia entre ejecutar instrucciones una sola vez al encender la placa y ejecutarlas en un ciclo infinito de repetición constante.

Comprenderemos cómo la modificación de los tiempos de espera afecta directamente a la percepción de velocidad de una animación en pantalla.

Realización

Paso 1: Configuración de la mesa de trabajo

Entrad en Tinkercad, cread un proyecto nuevo de circuito con la tarjeta micro:bit y abrid la interfaz de programación por bloques.

Paso 2: Identificar el bucle infinito

Como hemos podido intuir del proyecto anterior, todo el código que coloquemos en el interior del bloque `siempre` se repetirá en bucle continuo mientras la placa reciba energía.

Paso 3: Construir el primer fotograma (Corazón grande)

Arrastrad dentro del bloque por siempre el bloque `mostrar icono` y seleccionad la figura del corazón.

Para que el ojo humano pueda apreciar este primer dibujo antes de que cambie, añadid justo debajo un bloque `esperar 1 segundos`.

Paso 4: Construir el segundo fotograma (Corazón pequeño)

A continuación, debajo del tiempo de espera, colocad el bloque `mostrar leds`. En lugar de elegir una figura ya hecha, haced clic sobre los cuadros de la cuadrícula de 5x5 para encender manualmente los LEDs que forman la silueta de un corazón pequeño.

Añadid justo debajo otro bloque `esperar 1 segundos`. Esta segunda espera es imprescindible; si no la colocamos, la tarjeta volverá de forma instantánea al inicio del bucle `siempre` sin darnos tiempo a percibir en la pantalla el dibujo que acabamos de diseñar.

Paso 5: Añadir respuestas emocionales con los botones

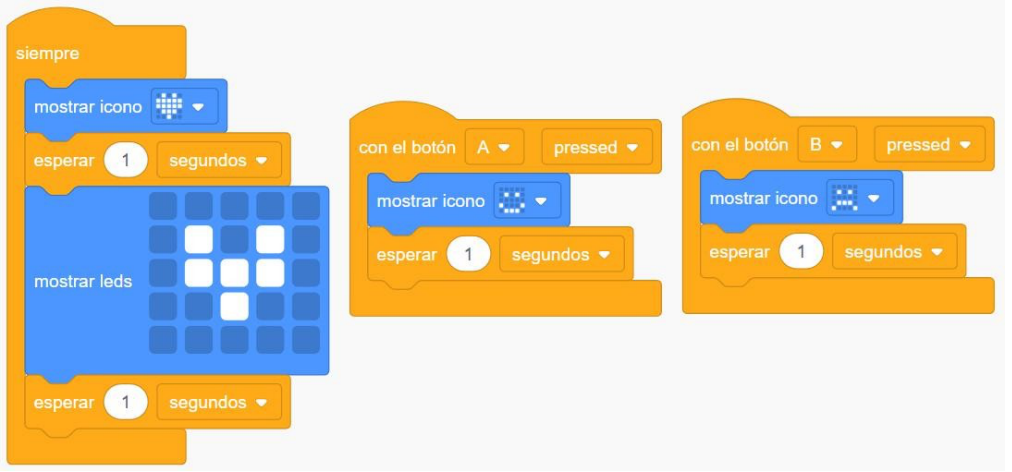
Vamos a hacer que la micro:bit cambie su estado de ánimo según el botón que pulsemos:

Añadid el bloque `con el botón A pressed`. Dentro de él, colocad un bloque mostrar icono con una carita feliz.

Añadid el bloque `con el botón B pressed`. Dentro de él, colocad un bloque mostrar icono con una carita triste.

Paso 6: Simulación y prueba de ritmo

Iniciad la simulación en Tinkercad. Observad cómo, de forma automática, el corazón de la micro:bit empieza a pulsar con un ritmo constante. Probad a interactuar con la placa utilizando las instrucciones programadas en los botones para comprobar los cambios que se producen en la matriz de LEDs.



Repaso

- El bloque `siempre` se denomina bucle. Es una estructura de control que ejecuta sin interrupción el código contenido en su interior, volviendo a empezar automáticamente indefinidamente.
- Por otro lado, el bloque `mostrar leds` nos da libertad absoluta para diseñar cualquier forma encendiendo o apagando los leds de manera individual, pudiendo formar la imagen que se nos antoje.
- La sensación de movimiento se logra mostrando una secuencia de imágenes estáticas intercaladas con pausas de tiempo adecuadas. El secreto reside en mantener un tiempo equilibrado entre cada imagen. Si comprobamos que nuestra animación parece congelada, es porque la pausa es demasiado alta; si, por el contrario, parece acelerada o no se aprecia el efecto, es demasiado baja. Os sugiero probar tiempos de entre 200 y 500 milisegundos para lograr un latido natural.

Desafío

¿Seríais capaces de hacer que el corazón no solo lata, sino que realice un efecto de giro o rotación cuando el usuario interactúe con la placa?

Modificad el bloque `con el botón A pressed` para que, en lugar de mostrar una cara fija, ejecute una pequeña animación donde el corazón parezca girar sobre su propio centro. Haced los distintos fotogramas utilizando el bloque `mostrar leds` y una pequeña pausa después de cada uno de ellos. Pensad antes cuántos diseños de `mostrar leds` distintos necesitaríais encadenar y qué tiempo en milisegundos deberíais colocar entre cada uno para que el giro se aprecie con fluidez.



Haced varias pruebas hasta comprobar que el diseño consigue el efecto deseado.

Proyecto 4: ¿Cuánta gente ha venido a la fiesta?

Dificultad: ★★☆☆☆ Tiempo: 30 m

Aprendizaje

Aprenderemos a crear y gestionar variables, que son "cajas de memoria" para almacenar datos numéricos que van cambiando a lo largo del tiempo.

Descubriremos cómo sumar valores a una variable almacenada para llevar el control de aforo en tiempo real.

Comprenderemos la importancia de inicializar los datos a cero al arrancar un sistema para asegurar que las lecturas de entrada sean exactas.

Realización

Paso 1: Entorno de trabajo

Acceded a Tinkercad, cread un nuevo proyecto de circuito con la tarjeta micro:bit y abrid la interfaz en modo de programación por "Bloques".

Paso 2: Crear la caja de memoria (Variable)

Imaginad que estáis en la puerta de una fiesta y queréis contar las personas que entran. Para guardar ese dato en la tarjeta, necesitamos crear una variable:

Id a la sección de bloques "● Variables" y haced clic en la opción "Crear variable...". Os saldrá una ventana emergente solicitando un nombre para nuestra nueva variable, llamadla "aforo". Veréis que se generan nuevos bloques específicos para esa variable.

Paso 3: Poner el contador a cero al abrir puertas

Al encender la tarjeta o comenzar el evento, el contador debe empezar siempre limpio desde el número cero:

Dentro del bloque `al iniciar`, colocad el bloque `definir aforo en 0`.

A continuación, dentro del bloque `siempre`, agregad un bloque `mostrar numero 0`. Sustituid el cero por el bloque ovalado con el nombre de vuestra variable: `aforo`.

Paso 4: Registrar un nuevo invitado con el botón A

Cada vez que entre una persona por la puerta, pulsaremos el botón A para sumarla al marcador:

Colocad en el tapiz de trabajo el bloque `con el botón A pressed`. Añadid dentro el bloque `cambiar aforo por 0`. Sustituid ese cero por un uno, que será la cantidad a sumar cada vez que pulséis el botón.

Paso 5: Vaciar el aforo con el botón B

Si la fiesta termina usaremos el otro botón para poner el contador a cero:

Arrastrad el bloque `con el botón B pressed`. Añadid en su interior el bloque `definir aforo en 0`.

Paso 6: Prueba en el simulador

Haced clic en "Iniciar simulación". Pulsad varias veces sobre el botón A como si estuvieran entrando invitados y observad cómo la pantalla matriz refleja la cifra actualizada. Pulsad el botón B para comprobar que el marcador se resetea al instante.



Repaso

- Las variables son contenedores de información dentro del código que nos permiten guardar, modificar y consultar datos mientras el programa se está ejecutando.
- Es imprescindible que a las variables les asignemos un valor de partida (habitualmente 0) dentro del bloque `al iniciar`.
- Mediante el bloque `definir aforo en 0` hemos dado un valor exacto a nuestra variable, y con `cambiar aforo por 1` modificamos la cantidad contenida dentro de la variable en el valor que se indique (en este caso sumamos uno).

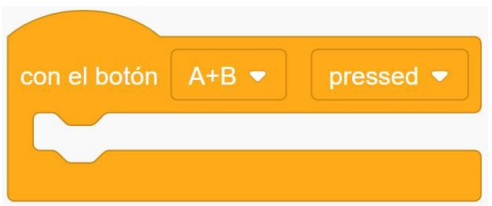
Desafío

¿Cómo podríais hacer que este control de acceso sea más completo y permita gestionar también a la gente que sale de la fiesta?

Intentad modificar el programa para que con el botón A presionado sumemos una persona al aforo, pero con el botón B presionado restemos una persona (cambiar la variable por -1).

La tarjeta micro:bit también admite que se reaccione de manera distinta si se pulsan simultáneamente ambos botones. Tinkercad simula esta opción añadiendo a su tarjeta un nuevo botón cuando iniciamos la simulación.

Como ahora el botón B está ocupado con otra acción, ¿qué combinación necesitaríais si quisierais utilizar la pulsación simultánea de los dos botones para reiniciar la cuenta a cero cuando se vacíe la sala?



Proyecto 5: ¡Lanza los dados!

Dificultad: ★★☆☆☆ Tiempo: 25 m

Aprendizaje

En esta ocasión, descubriremos cómo hacer que la tarjeta seleccione un valor numérico al azar dentro de un rango determinado, y también aprenderemos a programar la micro:bit para que reaccione al agitarla, utilizando su acelerómetro integrado.

Realización

Paso 1: Preparación de la mesa de trabajo

Entrad en Tinkercad, cread un nuevo proyecto de circuitos con la tarjeta micro:bit y abrid el editor en modo de programación por "Bloques".

Paso 2: Crear la variable para almacenar la tirada

Como el número que va a salir al agitar el dado cambia en cada lanzamiento, necesitamos guardarlo temporalmente en una caja de memoria:

Acceded a la sección de bloques "● Variables" y seleccionad "Crear variable...". A la variable llamadla "resultado".

Paso 3: Inicialización de la variable

Para asegurarnos de que la variable comienza limpia al arrancar el programa, dentro del bloque `al iniciar`, colocad el bloque `definir resultado en 0`.

Paso 4: Capturar el gesto de agitar la micro:bit

En lugar de presionar un botón, activaremos el dado moviendo la placa. En la categoría "● Entrada", elegid el bloque de evento `en shake`.

Dentro de este bloque, colocad `definir resultado en 0`, pero reemplazad el número 0 encajando en su lugar el bloque verde de la sección "● Matemáticas" llamado `seleccionar de forma aleatoria entre 1 y 10`. Para que el cambio se haga correctamente se ha de hacer coincidir el inicio redondeado de ambos bloques.

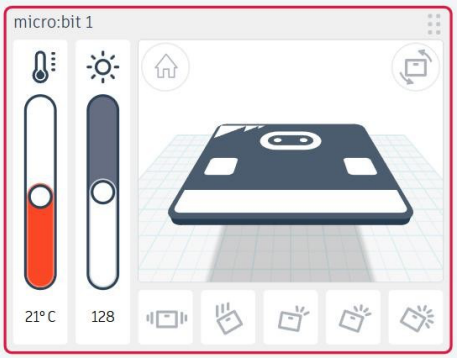
Cambiad ahora los valores del bloque para que escoja un número de 1 a 6.

Paso 5: Mostrar la cifra obtenida en pantalla

Para ver el número que nos ha tocado en la tirada, justo debajo de la asignación de la variable dentro del evento `en shake`, colocad el bloque `mostrar número 0`. Sustituid el cero por el bloque ovalado de la variable resultado.

Paso 6: Simulación en Tinkercad

Haced clic en "Iniciar simulación". Como no es posible realizar directamente algunas acciones sobre la tarjeta virtual micro:bit que nos aparece en Tinkercad, os habréis dado cuenta de que cuando iniciamos la simulación sale debajo de la tarjeta un menú de acciones como el de la imagen.



Investigad las distintas acciones que nos permite realizar, y descubrid cuál es la que agita la tarjeta.

Hacedlo varias veces y comprobad cómo la pantalla muestra una cifra distinta del 1 al 6 seleccionada al azar en cada movimiento.



Repaso

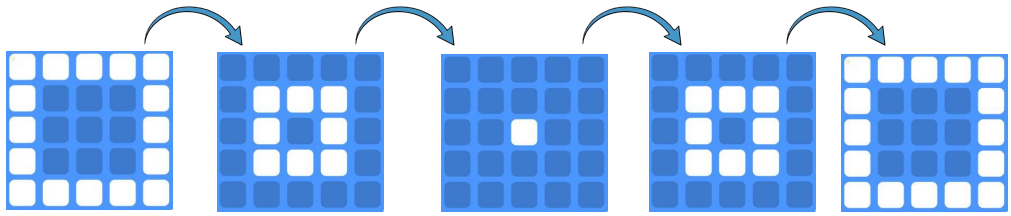
- La tarjeta micro:bit dispone de un acelerómetro, que es un sensor capaz de detectar la inclinación y los movimientos de la placa en el espacio, como cuando se agita, gira, ...
- Los sistemas informáticos disponen de funciones de aleatoriedad que permiten a los programas elegir un número cualquiera dentro de un intervalo. Cuando trabajéis con generación de números aleatorios es vital aseguraros de que los límites están bien fijados. Si ponéis el rango del 1 al 6, el sistema elegirá cualquiera de esas 6 posibilidades con exactamente la misma probabilidad. Probad a agitar la placa varias veces seguidas para comprobar que los números van cambiando y no se repiten siempre en la misma secuencia.
- Lectura de variables en pantalla: Combinar el bloque mostrar número con una variable nos permite visualizar el dato guardado en la memoria en tiempo real.

Desafío

Como habéis comprobado, el resultado sale inmediatamente después de agitar la tarjeta, pero ¿cómo podríais hacer que la micro:bit avise visualmente de que está "pensando" antes de enseñar la cifra definitiva?

Intentad modificar el bloque de evento `en shake` para que, justo antes de mostrar el número final, la pantalla haga una pequeña animación rápida.

Recordad que es importante que hagáis una pausa entre fotograma y fotograma.



Proyecto 6: ¡Qué calor hace!

Dificultad: ★★☆☆☆ Tiempo: 30 m

Aprendizaje

Aprenderemos a capturar datos del entorno real utilizando el sensor de temperatura integrado en el propio procesador de la micro:bit, y guardaremos esas lecturas continuas en una variable para que tome decisiones y emita avisos visuales según la temperatura ambiente supere un determinado umbral.

Realización

Paso 1: Configuración de la mesa de trabajo

Entrad en Tinkercad, cread un nuevo proyecto de circuitos con la tarjeta micro:bit y abrid la interfaz de programación en modo de "Bloques".

Paso 2: Crear la variable

Para poder trabajar con la temperatura que mide la tarjeta, hemos de almacenar esa información en una variable. Dirigíos a la categoría de bloques "● Variables" y seleccionad "Crear variable...". Nombrad la variable como "actual".

Paso 3: Inicializar la variable al arrancar

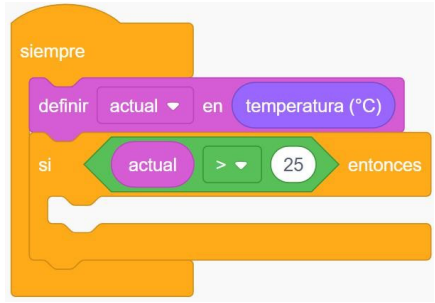
En el bloque `al iniciar`, nos aseguraremos de inicializar la variable, añadiendo el bloque `definir actual en 0`.

Paso 4: Capturar la temperatura en el bucle principal

Para que tome la temperatura continuamente, colocad dentro del bucle `siempre` el bloque `definir actual en 0`. Reemplazad el número 0 encajando en su lugar el bloque ovalado `temperatura (°C)` que encontraréis dentro de la sección "● Entrada".

Paso 5: Tomar decisiones según el nivel de calor

En esta ocasión no vamos a mostrar el valor de la variable, sino que vamos a hacer que se nos muestre un mensaje si hace demasiado calor. Para ello, debajo de la línea anterior, dentro de siempre, añadid un bloque condicional `si verdadero entonces ...`. En la casilla verde de la condición, encajad un bloque comparador de la categoría "● Matemáticas" con el símbolo "=" que tendremos que cambiar por el de mayor que (>). En el lado izquierdo del comparador colocad vuestra variable "actual", y en el lado derecho escribid un umbral límite, por ejemplo 25.



En el interior de la sección "entonces", colocad un bloque `mostrar icono` con una carita triste.

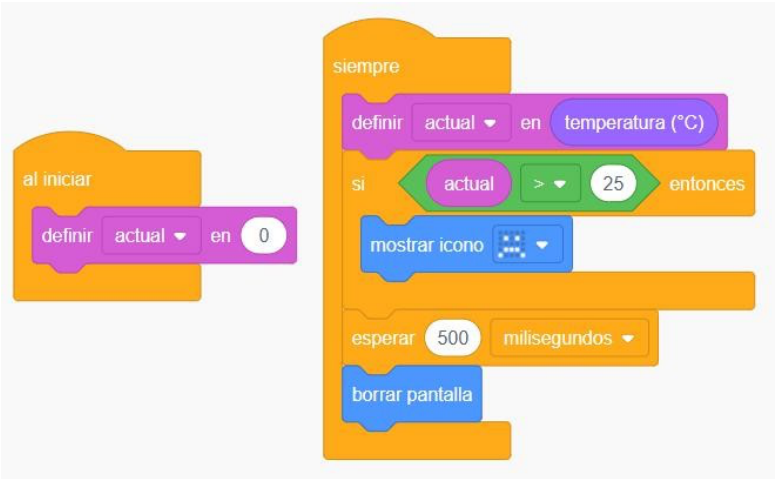
Ahora, fuera del condicional `si ... entonces ...` pero dentro del bucle `siempre`, añadid una pausa de 500 milisegundos y un bloque `borrar pantalla`.

Paso 6: Prueba y simulación en Tinkercad



Haced clic en "Iniciar simulación". Igual que en el proyecto anterior nos aparece debajo de la tarjeta un menú de acciones. Si os fijáis, a la izquierda, aparece una barra deslizante para simular la temperatura exterior. Variadla para ver cómo responde la tarjeta a temperaturas superiores o inferiores al umbral de 25 que pusisteis.

Consejo: Cuando probéis en la tarjeta física real, el sensor mide la temperatura interna del chip. Si sostenéis la placa presionando el procesador con los dedos durante unos segundos, veréis cómo la temperatura registrada empieza a subir gradualmente en la pantalla.



Repaso

- El sensor de temperatura integrado nos devuelve un valor numérico en grados Celsius (°C) mediante el bloque de entrada `temperatura (°C)`.
- Un condicional como `si ... entonces ...` permite ejecutar un código solo cuando se cumple una condición.
- Llamaremos "umbral de activación" a la cifra que hace que se cumpla o no la condición.

Desafío

¿Cómo podríais hacer que este termómetro tenga dos alertas independientes: una para cuando hace demasiado calor y otra para cuando hace demasiado frío?

Podéis añadir dentro del bucle `siempre` un segundo bloque condicional `si ... entonces ...` independiente. Utilizad en él un comparador menor que (<) para comprobar si la variable "actual" cae por debajo de los 15 °C. ¿Qué dibujo podríais colocar dentro de ese segundo condicional para indicar que hace frío? Quizá os interese cambiar también el icono cuando hace calor.

Proyecto 7: ¡Abre las cortinas!

Dificultad: ★★☆☆☆ Tiempo: 30 m

Aprendizaje

En este proyecto veremos cómo la micro:bit aprovecha sus LEDs también para medir la intensidad de la luz ambiental. Esta intensidad se trabaja con un rango de lectura que abarca desde 0 (oscuridad) hasta 255 (luz máxima).

De manera similar a como se hizo en el proyecto anterior, programaremos la tarjeta para evaluar un umbral de luz pero, en este caso, puede tomar dos caminos distintos: ejecutar una alerta si no hay suficiente claridad o apagar la pantalla cuando las condiciones sean óptimas.

Realización

Paso 1: Preparación del entorno de trabajo

Acceded a Tinkercad, cread un nuevo proyecto de circuitos con la tarjeta micro:bit y abrid el editor en modo de programación por "Bloques".

Paso 2: Crear la variable de luminosidad

Para ir guardando las mediciones de luz que captura la placa en cada instante, necesitamos una variable. Dirigíos a la categoría "● Variables" y elegid la opción "Crear variable...". Nombrad la variable como "luminosidad".

Paso 3: Inicialización al arrancar

En el bloque `al iniciar`, nos aseguraremos de inicializar la variable, añadiendo el bloque `definir luminosidad en 0`.

Paso 4: Muestreo continuo del nivel de luz

Queremos que la tarjeta esté midiendo la claridad del entorno constantemente en el bucle principal, por lo que dentro del bucle `siempre`, añadid el bloque definir luminosidad en 0. Reemplazad la casilla del 0

encajando en su lugar el bloque ovalado `nivel de luz` que encontraréis en la sección "● Entrada".

Paso 5: Programar el control de iluminación con condicional compuesto

Vamos a hacer que la micro:bit reaccione según la claridad que detecte utilizando una sola estructura de decisión de doble vía. Buscad en la sección de bloques "● Control" el bloque `si verdadero entonces ... si no ...` y encajadlo dentro de `siempre`, justo debajo de la asignación de la variable.

Reemplazad la casilla `verdadero` encajando un bloque comparador de la sección "● Matemáticas" con el operador menor que (<). Recordad que primero aparece con el operador igual (=), pero que lo debéis modificar.

En el lado izquierdo del comparador colocad vuestra variable "luminosidad", y en el lado derecho escribid un umbral límite de penumbra, por ejemplo 80.

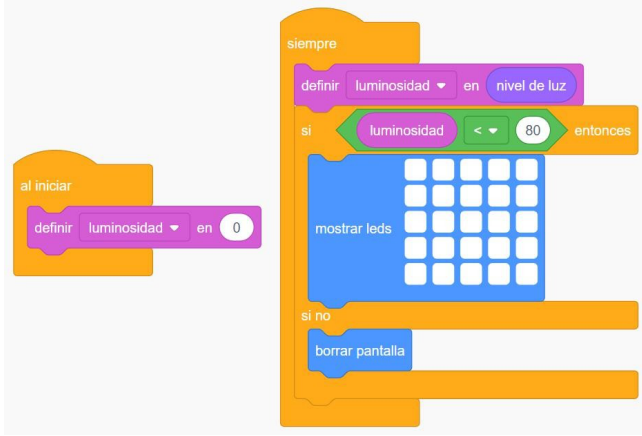
En el bloque interno de `entonces` (poca luz): Añadid el bloque `mostrar leds` con todos los LEDs encendidos, a modo de linterna para avisar de que falta luz en la estancia.

En el bloque interno de `si no` (luz suficiente): Colocad la orden `borrar pantalla` para que los LEDs permanezcan apagados mientras la habitación esté iluminada.

Paso 6: Simulación e inspección en Tinkercad

Haced clic en "Iniciar simulación". En Tinkercad, debajo de la placa micro:bit, veréis aparecer un regulador deslizante con el dibujo de un sol. Arrastrad el mando hacia abajo (simulando penumbra) y comprobad cómo, al bajar de 80, la pantalla se enciende. Arrastradlo hacia arriba (más de 80) y observaréis cómo la pantalla se apaga automáticamente gracias a la sección `si no`.





Repaso

- El sensor de nivel de luz mide la cantidad de claridad que incide sobre la tarjeta y la traduce en una cifra numérica continua entre 0 y 255 usando el bloque nivel de luz. Cuando probéis en la tarjeta física real, tened en cuenta que la sombra de vuestra propia mano al acercaros a la placa puede hacer bajar la lectura bruscamente.
- El condicional `si verdadero entonces ... si no ...` permite diversificar el código en dos caminos alternativos y excluyentes: ejecuta unas acciones cuando la regla se cumple y otras totalmente distintas cuando es falsa.
- El bloque `definir luminosidad en 0` dentro de `al iniciar` garantiza una puesta a cero correcta.

Desafío

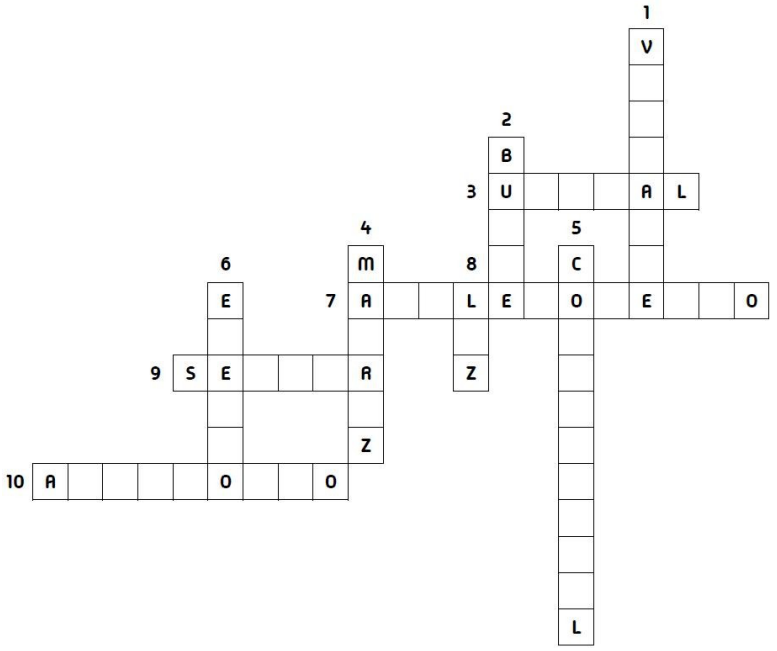
Intentad modificar el programa para que funcione al revés: la pantalla debe estar apagada mientras haya oscuridad total, pero si entra luz repentinamente (la variable "luminosidad" supera el valor 150), el programa debe activar un icono parpadeante. ¿Qué operador matemático deberíais cambiar dentro del condicional?

Gimnasia mental I: Crucigrama

Antes de avanzar al siguiente bloque de proyectos, resolved el siguiente crucigrama para repasar los conceptos clave que habéis aprendido sobre la micro:bit y la programación.

Leed con atención las pistas, deducid la palabra correcta y escribidla letra a letra en la cuadrícula correspondiente.

Las soluciones las encontraréis en las últimas páginas del libro.



Horizontales

3. Cifra o límite numérico fijado en una condición lógica para activar una respuesta o alerta cuando la medida de un sensor lo sobrepasa o queda por debajo.
7. Sensor interno de la tarjeta capaz de detectar movimientos, inclinaciones o sacudidas.
9. Componente capaz de medir magnitudes físicas del entorno real, como la temperatura ambiental o la intensidad de iluminación.
10. Valor numérico obtenido al azar dentro de un rango determinado.

Verticales

1. Casilla de memoria etiquetada con un nombre que permite guardar y actualizar datos durante la ejecución de un programa.
2. Estructura de programación que repite constantemente las instrucciones situadas en su interior.
4. Conjunto organizado de 25 LEDs colocados en una cuadrícula de 5x5 que forman la pantalla principal de la micro:bit.
5. Estructura lógica (si verdadero... entonces... si no) que permite a la tarjeta tomar decisiones ejecutando un camino u otro.
6. Suceso o acción externa detectable (como presionar un botón o agitar la placa) que desencadena de forma inmediata un bloque de código específico.
8. Magnitud física ambiental que la micro:bit puede medir utilizando su propia pantalla de diodos en una escala de 0 a 255.

Proyecto 8: Te desafío, ¿piedra, papel o tijera?

Dificultad: ★★☆☆☆ Tiempo: 35 m

Aprendizaje

Aprenderemos a asociar valores numéricos elegidos al azar (1, 2 y 3) con opciones reales de un juego mediante el código. Para ello utilizaremos estructuras condicionales anidadas para evaluar las tres opciones posibles que puede tomar una variable.

Realización

Paso 1: Preparación del escenario de trabajo

Entrad en Tinkercad, cread un nuevo proyecto de circuitos con la tarjeta micro:bit y abrid la ventana de edición en modo de programación por "Bloques".

Paso 2: Crear la variable

Id a la categoría Variables y seleccionad la opción "Crear variable...". Llamadla "elección".

Paso 3: Inicialización al arrancar

Dentro de `al iniciar`, añadid el bloque `definir elección en 0`.

Paso 4: Generar la jugada mediante el movimiento

Activaremos el lanzamiento del juego cada vez que agitemos la micro:bit, por lo que de la sección "● Entrada", elegid el bloque de evento `en shake`.

En su interior, colocad la orden `definir elección en 0`. Reemplazad la casilla del 0 encajando el bloque de la categoría "● Matemáticas" llamado `seleccionar de forma aleatoria entre 1 y 10`. Modificad los valores para que seleccione un número del 1 al 3 (donde 1 será Piedra, 2 será Papel y 3 será Tijera).

Paso 5: Diseñar las respuestas visuales según la variable

Ahora debemos evaluar la cifra que ha salido al azar y mostrar el icono correspondiente en la pantalla LED, así que dentro del evento `en shake` (justo debajo de la asignación aleatoria), colocad un bloque condicional `si verdadero entonces ... si no ...`.

En la parte `si` de la condición, encajad un bloque comparador de la categoría "● Matemáticas" con el símbolo de igualdad (=). Comprobad si la variable "elección" es igual a 1.



En este caso el valor 1 sería Piedra, por lo que dentro del bloque, añadid un bloque `mostrar leds` y diseñad un cuadrado pequeño de 3x3 LEDs centrados que simule una piedra.

Dentro del bloque `si no`, encajad un nuevo condicional `si verdadero entonces ... si no ...`, es decir, va uno dentro de otro. En la parte `si` de esta nueva condición, haced una comparación similar a la anterior, pero comparad si la variable "elección" es igual a 2.

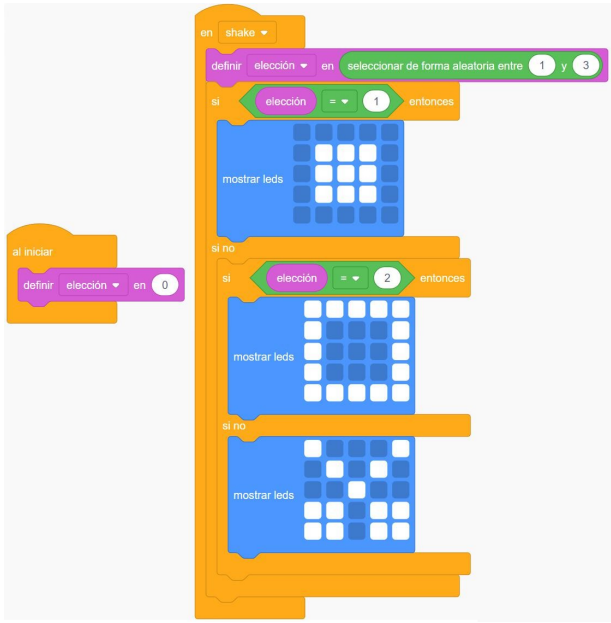
Para simular la elección de Papel, en el apartado `entonces`, colocad un bloque `mostrar leds` encendiendo el marco exterior de 5x5 LEDs.

En el interior del último bloque `si no`, únicamente nos queda por representar la opción de Tijera, por tanto colocad un bloque `mostrar leds` y haced un dibujo que se asemeje a una tijera.



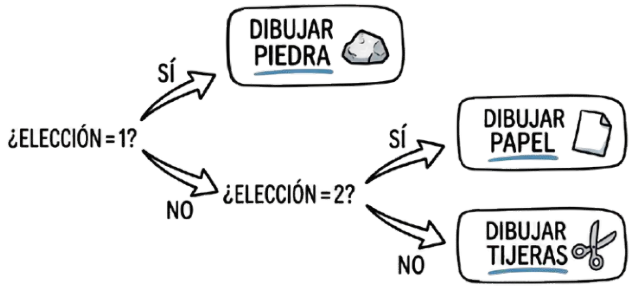
Paso 6: Simulación y prueba del juego

Haced clic en "Iniciar simulación". Al simular la agitación de la placa micro:bit en Tinkercad, veréis cómo en cada sacudida la pantalla muestra de forma inmediata y aleatoria una piedra, un papel o unas tijeras para jugar contra vosotros.



Repaso

- La tarjeta ha generado un número aleatorio del 1 al 3 y los hemos convertido en imágenes con significado dentro del juego.
- Lo más importante del proyecto es que entendáis que los condicionales anidados nos permiten resolver decisiones de más de dos posibilidades.



Desafío

¿Podríais añadir una cuenta atrás antes de revelar la jugada?

Intentad modificar el evento `en shake` para que, justo antes de elegir el número aleatorio, la tarjeta muestre la secuencia "1... 2... 3..." con su correspondiente pausa entre números.

Proyecto 9: ¡10.000 pasos!

Dificultad: ★★☆☆☆ **Tiempo:** 30 m

Aprendizaje

Con este proyecto aprenderemos a utilizar el sensor de aceleración integrado en la tarjeta para capturar sacudidas o impulsos físicos equivalentes a un paso, e ir acumulando el valor en una variable.

Realización

Paso 1: Preparación del escenario de trabajo

Entrad en Tinkercad, cread un nuevo proyecto de circuitos y abrid la ventana de edición en modo de programación por "Bloques".

Paso 2: Crear la variable pasos

Para llevar el registro acumulado de los pasos que va dando el usuario, hay que crear una variable. Id a la categoría "● Variables" en Tinkercad y seleccionad la opción "Crear variable...". Llamad a la variable "pasos".

Paso 3: Inicializar la variable

En el bloque `al iniciar`, colocad el bloque `definir pasos en 0`.

Paso 4: Detectar cada paso con el acelerómetro

Imaginad que tenéis la tarjeta micro:bit enganchada en uno de vuestros zapatos. Queremos que la tarjeta sume un paso cada vez que detecte un movimiento brusco al caminar. Para ello seleccionad en la categoría "● Entrada" el bloque de evento `en shake`.

Dentro de este bloque, colocad la orden `cambiar pasos por 0`, pero reemplazad el 0 por un 1.

Paso 5: Programar el botón de reinicio

Para poner el contador a cero seleccionad en la categoría "● Entrada" el

bloque `con el botón A pressed`, encajad en su interior la orden `definir pasos en 0`. Colocad también el bloque `mostrar número` con la variable `pasos` para confirmar visualmente al usuario que la cuenta vuelve a ser 0. Añadid una pausa con el bloque `esperar 1 segundos` y continuad con un `borrar pantalla`.

Paso 6: Mostrar la cantidad de pasos

Para mostrar el número de pasos seleccionad en la categoría "● Entrada" el bloque `con el botón B pressed`, y colocad en su interior el bloque `mostrar número` con la variable `pasos`. Añadid una pausa con el bloque `esperar 1 segundos` y continuad con un `borrar pantalla`.

Paso 7: Simulación y comprobación en Tinkercad

Haced clic en "Iniciar simulación". En Tinkercad, simulad las sacudidas del andar y observad, pulsando el botón B, cómo el contador de pasos se incrementa. Después, pulsad el botón A virtual para comprobar que el podómetro se pone a cero correctamente.



Desafío

¿Cómo podríais hacer para que os felicite al alcanzar una meta de pasos?

Intentad añadir dentro del evento `en shake` una estructura condicional `si verdadero entonces` que compruebe si la variable "pasos" ha llegado a un número determinado. Si se alcanza la meta, haced que aparezca una imagen de una copa en la pantalla LED.

Proyecto 10: ¡Qué fuerza tienes!

Dificultad: ★★☆☆☆ Tiempo: 35 m

Aprendizaje

Capturaremos la intensidad exacta de una sacudida utilizando el acelerómetro integrado para obtener valores medidos en mili-fuerza G (mg).

Trabajaremos con dos variables, una para almacenar la fuerza ejercida actualmente con otra que almacena el récord.

Aprenderemos que los simuladores pueden tener límites.

Realización

Paso 1: Crear las variables de medición y de marca máxima

Para poder almacenar tanto el valor de la sacudida actual como el récord alcanzado durante la sesión, crearemos dos variables: "fuerza" y "récord".

Paso 2: Inicializar los datos al encender la tarjeta

En el bloque `al iniciar`, colocad las órdenes `definir fuerza en 0` y `definir récord en 0`.

Paso 3: Capturar la intensidad del impacto

Seleccionad en la categoría "● Entrada" el bloque de evento `en shake`. En su interior, asignad el valor del movimiento a nuestra variable con `definir fuerza en 0`, pero sustituyendo el 0 por el bloque `aceleración (mg) strength` de la sección "● Entrada" .

Paso 4: Comprobar el récord con un condicional de condición compuesta

Ahora evaluaremos si la sacudida actual supera el récord guardado. Para ello, dentro del evento `en shake`, colocad el bloque condicional `si verdadero entonces`. Reemplazad la casilla `verdadero` por un bloque `verdadero y verdadero` de la categoría "● Matemáticas".

En el lado izquierdo del operador Y, evaluad si "fuerza" > "récord". Para ello, sustituid el primer `verdadero` por la casilla `1 = 1`, sustituyendo ahora el primer 1 con la variable `fuerza` y el segundo 1 con la variable `récord`. El símbolo "=" sustituidlo por ">"

En el lado derecho del operador Y, evaluad si fuerza > 1200 (este límite evita que movimientos suaves de la tarjeta cuenten como un impacto válido).

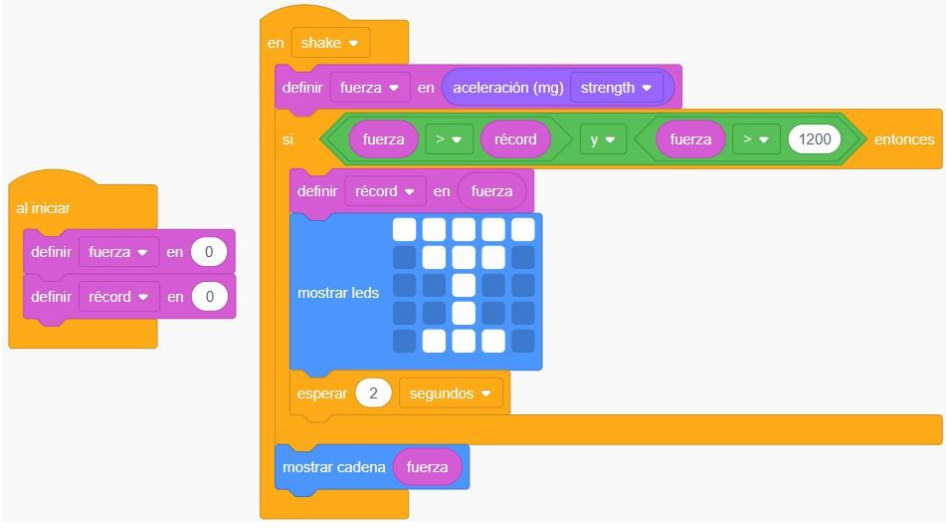
En la sección entonces (cuando ambas condiciones son verdaderas a la vez), guardad el nuevo récord con la orden `definir récord en 0` sustituyendo el 0 por la variable `fuerza` y mostrad un icono festivo como el de una copa durante 2 segundos.

Paso 5: Mostrar la fuerza ejercida

A continuación del `si verdadero entonces`, pero dentro de `en shake`, mostraremos el valor de la sacudida, almacenado en la variable `fuerza`. Para ello insertad el bloque `mostrar cadena fuerza`.

Paso 6: Simulación y comprobación en Tinkercad

Haced clic en "Iniciar simulación" y pulsad en el icono de agitar. Notaréis que siempre la agitación la hace con una fuerza similar a 1000, por lo que nunca se batirá el récord. Este proyecto funcionará bien con la tarjeta física.



Proyecto 11: No pierdas el Norte

Dificultad: ★★☆☆☆ Tiempo: 30 m

Aprendizaje

Descubriremos cómo la placa micro:bit mide el campo magnético terrestre mediante la instrucción `rumbo de la brújula (°)` del bloque "● Entrada", la cual devuelve la dirección en grados respecto al Norte.

También aprenderemos a combinar varias condiciones dentro de una misma estructura condicional para que la tarjeta tome decisiones complejas únicamente cuando se cumplan simultáneamente varios requisitos (operador Y) o al menos uno de ellos (operador O).

Realización

Paso 1: Crear la variable de lectura magnética

Para consultar y procesar con fluidez la dirección hacia la que apunta la tarjeta, guardaremos dicho valor en una variable. Llamadla "rumbo".

Paso 2: Inicializar la tarjeta y calibrar el sensor

En el bloque `al iniciar`, colocad la instrucción `definir rumbo en 0`.

Paso 3: Capturar la orientación en el bucle principal

En el bloque `siempre`, colocad el bloque `definir rumbo en 0`. Sustituid el valor 0 por el bloque `rumbo de la brújula (°)`, que se encuentra en la categoría "● Entrada".

Paso 4: Clasificar los puntos cardinales mediante condicionales

Queremos que en la pantalla nos aparezca N (Norte), S (Sur), E (Este) u O (Oeste) en función de los grados almacenados en `rumbo`.

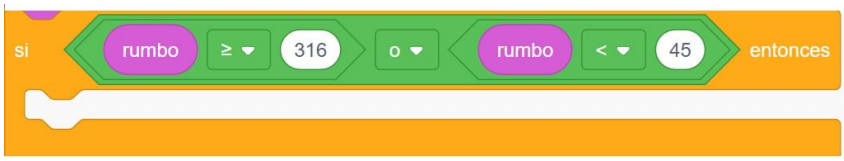
Añadid dentro de `siempre` un bloque condicional `si verdadero entonces`. Para el Norte, sustituid `verdadero` por un bloque `verdadero`

`y verdadero` de la categoría "● Matemáticas".

Sustituid el operador "Y" por el operador "O".

En el lado izquierdo del operador "O", evaluad si "rumbo" es mayor o igual al valor 316. Para ello, sustituid el primer `verdadero` por la casilla `1 = 1`, sustituyendo ahora el primer 1 con la variable `rumbo` y el segundo 1 con el valor 316. El símbolo "=" sustituidlo por "≥".

En el lado derecho del operador "O" realizamos una operación similar, sustituid el segundo `verdadero` por la casilla `1 = 1`, sustituyendo ahora el primer 1 con la variable `rumbo` y el segundo 1 con el valor 45. El símbolo "=" sustituidlo por "<".



En su interior poned el bloque `mostrar cadena` con la letra "N".

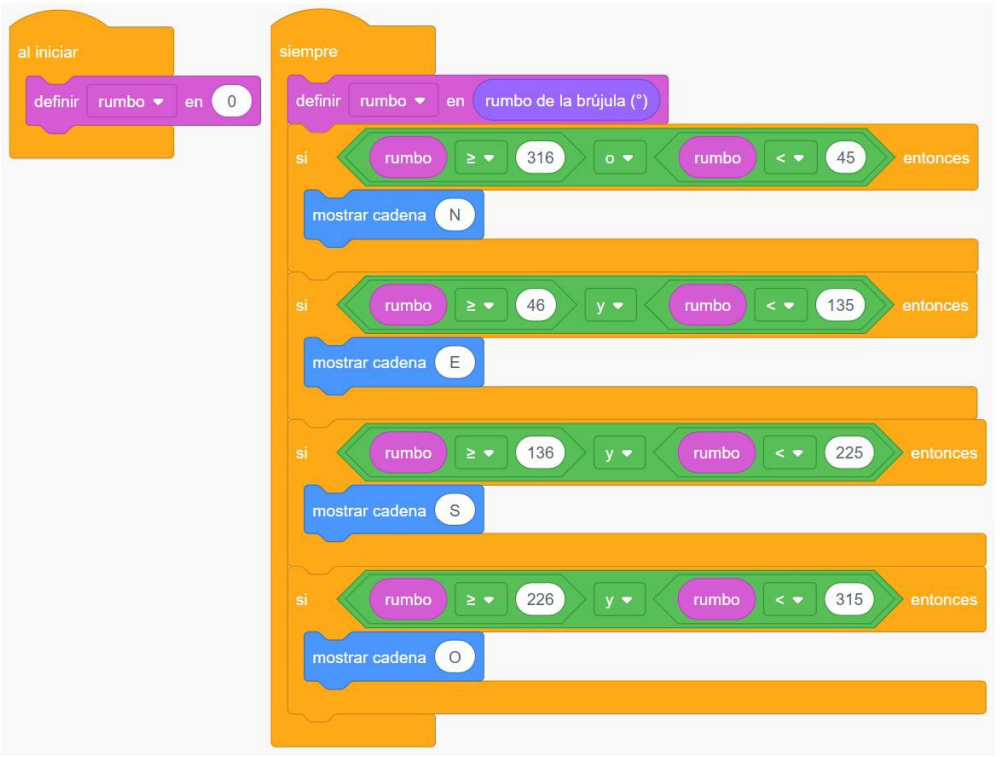
Repetid este paso 4 para el resto de puntos cardinales. Tened en cuenta que los grados varían para cada uno de ellos, así como el operador "Y" u "O". Lógicamente, también el texto a mostrar en cada ocasión será distinto. Ayudaos de la siguiente imagen.



Paso 5: Simulación y comprobación en Tinkercad

Haced clic en "Iniciar simulación". En el entorno de Tinkercad, observaréis que al ejecutar el programa aparece un control giratorio sobre la micro:bit virtual que simula la rotación física de la placa. Moved ese dial y comprobad cómo la pantalla cambia mostrando "N", "E", "S" u "O".

Si cargáis el código en la micro:bit física, notaréis que la primera vez que se ejecuta aparece un mensaje pidiendo llenar un círculo de puntos, es la calibración. Debéis inclinar la tarjeta en todas direcciones hasta rellenar todos los LED.



Repaso

- Este proyecto ha sido todo un desafío. Aprendimos que el magnetómetro es el sensor encargado de detectar las líneas del campo magnético terrestre y traducirlas a un valor numérico entre 0º y 359º.
- Un bloque condicional puede tener varias condiciones dentro de un mismo bloque, estando unidas con un operador "Y" si se han de cumplir las dos condiciones para que se realicen sus acciones, o con un operador "O" si basta con que al menos una de ellas se cumpla.

Proyecto 12: Tres... Dos... Uno... ¡Ya!

Dificultad: ★★☆☆☆ Tiempo: 35 m

Aprendizaje

Con el cronómetro de cuenta atrás que vamos a realizar, aprenderemos a programar la tarjeta para que el usuario pueda configurar libremente un parámetro de partida (en este caso, los segundos) sumando o restando unidades mediante los botones individuales antes de iniciar una tarea.

Utilizaremos la pulsación simultánea de los dos botones para desencadenar la rutina de la cuenta atrás sin interferir con la configuración del tiempo.

Diseñaremos una secuencia que vaya reduciendo el tiempo segundo a segundo y mostrando el valor en pantalla hasta que el contador alcance la cifra cero.

Realización

Paso 1: Crear las variables del temporizador

Para controlar tanto el valor programado por el usuario como el funcionamiento de la cuenta atrás, crearemos dos variables: "máximo" y "contador".

Paso 2: Fijar el valor inicial al arrancar

En el bloque de inicio prepararemos la tarjeta estableciendo un tiempo predeterminado de partida para que la pantalla no aparezca vacía. En el bloque `al iniciar`, colocad la orden `definir máximo en 5` (así comenzaremos con una base de 5 segundos). Para iniciar la otra variable, insertad `definir contador en 0`.

Completad el bloque con `mostrar número máximo`.

Paso 3: Decrementar el tiempo del cronómetro

Permitiremos que el usuario reduzca el tiempo de la cuenta atrás mediante la pulsación del botón izquierdo. Para ello, seleccionad en la categoría "● Entrada" el bloque `con el botón A pressed`.

Añadid en su interior un condicional simple `si verdadero entonces` para evaluar si contador = 0 (así evitamos modificar el tiempo si la cuenta atrás ya ha comenzado).

Dentro de este condicional, colocad la orden `cambiar máximo por -1`.

Añadid la instrucción `mostrar número máximo` para ver reflejado el cambio inmediatamente en la matriz de LED.

Paso 4: Incrementar los segundos

Programaremos el botón derecho para aumentar el tiempo a cronometrar. Para ello, seguiremos exactamente igual el paso 3, pero donde habla del botón A indicamos `con el botón B pressed`, y para incrementar los segundos, en este caso pondremos `cambiar máximo por 1`.

Paso 5: Lanzar la cuenta atrás con la entrada combinada A+B

Iniciaremos la rutina de descuento de tiempo cuando se presionen ambos botones a la vez.

Seleccionad de la categoría "● Entrada" el bloque de evento `con el botón A+B pressed`.

Colocad en su interior un bucle `repetir mientras verdadero` de la categoría "● Control". Sustituid la casilla `verdadero` por la condición `1 = 1`, que teneís que cambiar por `contador < máximo`.

Ahora, dentro del bucle, colocad la orden `mostrar número 0`. Sustituid el 0 por `1 + 1` , y convertidlo en `máximo - contador` . Esto hace que los segundos aparezcan marcha atrás.

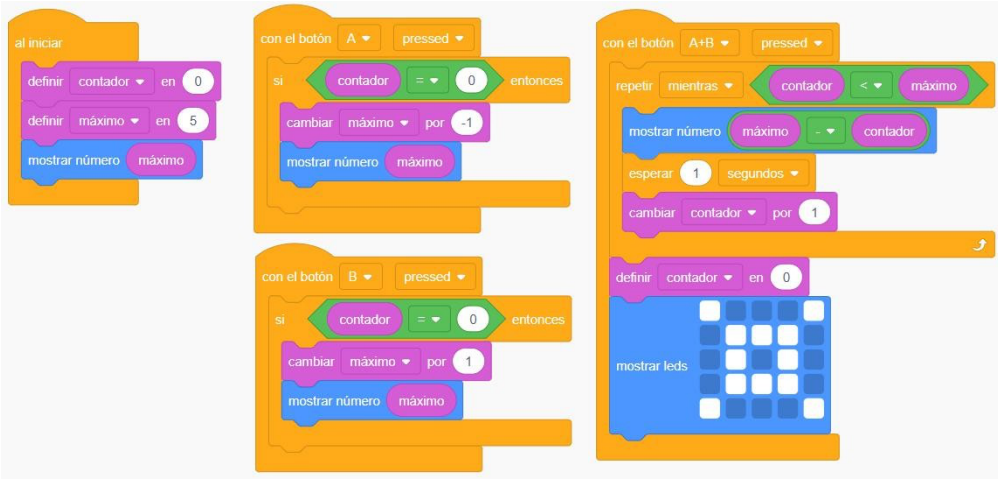
Inmediatamente después, dentro del bucle, incluid un bloque `esperar 1`

`segundos` y luego otro `cambiar contador por 1`.

Ya fuera del bucle de la cuenta atrás, poned el contador a cero con el bloque `definir contador en 0` y mostrad un icono final de aviso.

Paso 6: Simulación y comprobación en Tinkercad

Haced clic en "Iniciar simulación". Veréis que la pantalla muestra inicialmente el número 5. Haced clic en el botón A para reducir el valor de los segundos y en el botón B para aumentarlo. A continuación, pulsad sobre el botón virtual "A+B" que aparece en Tinkercad. Comprobad cómo la tarjeta va mostrando el descuento segundo a segundo de forma descendente y finaliza mostrando el icono elegido al llegar a cero.



Repaso

- Vimos como los botones nos sirven tanto para ajustar los datos (en nuestro caso los botones A y el B), como para ejecutar la tarea que se ha programado (pulsando ambos botones).
- Ante la imposibilidad de pulsar los dos botones en Tinkercad, nos aparece un botón virtual que simula esta acción.
- El bucle `repetir mientras` nos permite repetir un conjunto de órdenes únicamente mientras se cumpla una condición.

Gimnasia mental II: Wordoku

Haced una pausa para resolver un wordoku. Si no lo habéis visto antes, se trata de un juego de lógica, similar al sudoku. La regla es muy sencilla: ninguna letra puede repetirse dentro de una misma fila horizontal, columna vertical o caja de 3x3. Usa la lógica para descubrir qué letra falta en cada hueco.

E	N	C		T		D		R
D		I	E		N	K	T	
T		A	D	R		E		N
A		D	C	N	E	R	K	T
	T	N	A		K	I		E
	E	K			T	N	D	A
	D	T		E	A	C	R	
K		E	I		R		N	D
N	C	R	T	K		A		I

Rellena las casillas vacías con las letras de

T - I - N - K - E - R - C - A - D

Gimnasia mental III: Sopa de letras

Encuentra las palabras del conjunto que aparecen a la izquierda. Recuerda que estas están ocultas en horizontal, vertical o diagonal.

ACELERACIÓN
BUCLE
CALIBRACIÓN
CONDICIONAL
MAGNETÓMETRO
RUMBO
VARIABLE

J	Y	V	D	J	M	P	B	V	G	G	F	W	X
M	A	G	N	E	T	O	M	E	T	R	O	Q	H
Y	A	A	M	C	L	P	Z	J	B	C	K	Z	X
C	C	C	H	B	O	M	A	U	U	O	E	H	F
V	O	A	E	F	Q	X	F	P	C	U	Y	Y	H
R	N	L	G	L	X	T	I	L	L	W	E	H	L
L	D	I	V	B	E	N	H	U	E	F	B	G	B
W	I	B	I	A	T	R	Q	V	N	U	X	T	N
V	C	R	P	R	R	A	A	K	P	B	E	H	P
P	I	A	H	N	Q	I	B	C	L	H	S	J	T
B	O	C	I	L	F	V	A	D	I	X	M	E	M
W	N	I	G	Q	R	U	M	B	O	O	N	D	Z
C	A	O	W	D	C	S	G	C	L	O	N	R	Y
E	L	N	B	K	S	C	L	V	K	E	B	U	K

Proyecto 13: ¿Boca arriba o boca abajo?

Dificultad: ★★☆☆☆ Tiempo: 20 m

Aprendizaje

Descubriremos cómo interpretar las lecturas direccionales del acelerómetro mediante los bloques de eventos específicos de inclinación para saber si la superficie de la matriz de LED mira hacia el cielo o hacia el suelo.

Realización

Paso 1: Inicializar el sistema

En el bloque `al iniciar`, mostrad una carita sonriente mediante el bloque `mostrar icono` para indicar que la tarjeta está en posición correcta.

Debido a la sencillez de este proyecto, no es necesario incluir ningún bloque adicional.

Paso 2: Detectar la posición "cara arriba"

Programaremos el comportamiento que debe activar la micro:bit cuando la superficie de la placa mire hacia arriba. Seleccionad en la categoría "● Entrada" el bloque de evento `en shake`, y cambiamos `shake` por `screen up`.

En su interior, añadid un bloque `mostrar icono` que muestre una flecha señalando hacia arriba.

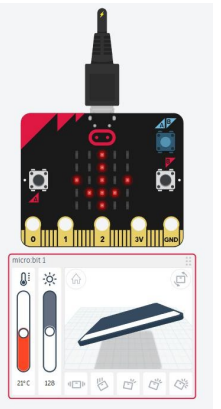
Paso 3: Detectar la posición "cara abajo"

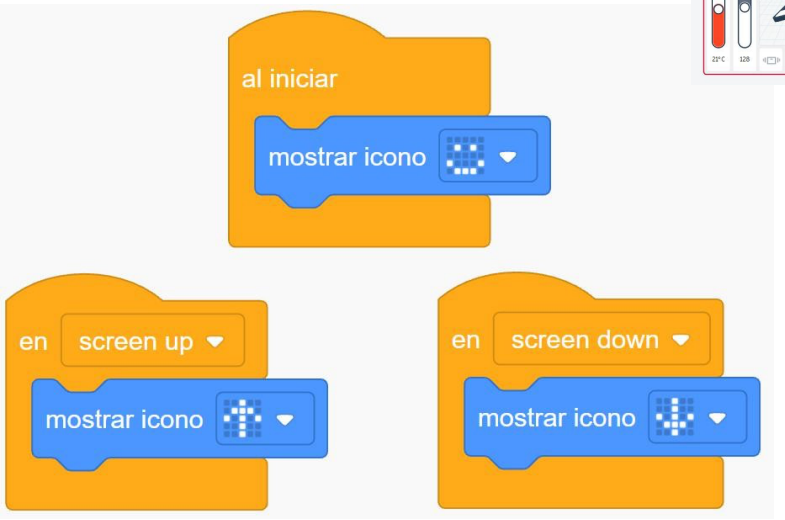
Repetid el paso anterior, pero con pequeños cambios: Seleccionad en la categoría "● Entrada" el bloque de evento `en shake`, y cambiamos `shake` por `screen down`.

En su interior, añadid un bloque `mostrar icono` que muestre una flecha señalando hacia abajo.

Paso 4: Simulación en Tinkercad

Haced clic en "Iniciar simulación". En la pantalla virtual de Tinkercad observaréis que manteniendo pinchada la micro:bit podréis rotarla en todas direcciones. Observad cómo el icono que se muestra varía según la posición de la tarjeta.





Repaso

- Los eventos de postura como `screen up` y `screen down` nos permiten saber cómo está posicionada la placa en el espacio físico.
- Utilizar iconos es importante para indicar en cada momento en qué situación se encuentra el programa. Una pantalla en negro nos puede desconcertar.

Desafío

¿Qué otros eventos de postura tiene Tinkercad?

Este proyecto ha sido muy sencillo, probad con el resto de eventos de postura, como `logo up`, `logo down`, `tilt left` y `tilt right`.

Proyecto 14: ¿Estás ahí?

Dificultad: ★★★☆☆ Tiempo: 40 m

Aprendizaje

Aprenderemos a configurar un canal de radio idéntico en dos placas para que puedan formar una red cerrada e intercambiar información sin cables.

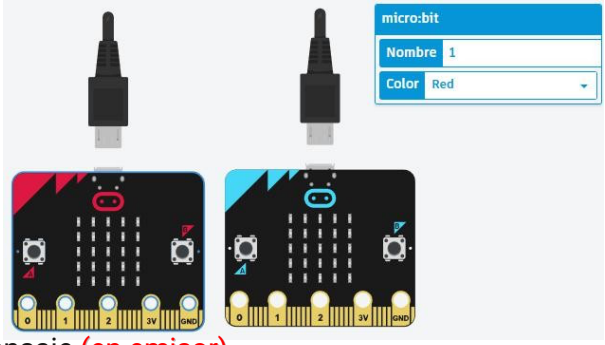
Descubriremos cómo empaquetar una variable o señal en la placa emisora para transmitirla por el aire.

Programaremos la placa receptora para que reaccione automáticamente mostrándonos en pantalla los datos enviados por la otra tarjeta.

Realización

Paso 1: Preparación de la mesa de trabajo

Entrad en Tinkercad, cread un nuevo proyecto de circuitos. En este caso, arrastrad dos tarjetas micro:bit a la zona de trabajo. Fijaos que, al pinchar sobre ellas, aparece un nombre distinto para cada una de ellas: 1 y 2. Cambiad esos nombres por: emisor y receptor. Cada tarjeta tendrá asociado un programa distinto, así que debemos tener seleccionada la tarjeta correcta.



Paso 2: Crear la variable de mensaje (en emisor)

Para controlar el dato que vamos a transmitir, crearemos una variable en la memoria: Id a la categoría "● Variables" y cread una llamada "mensaje".

Paso 3: Configurar el canal de radio al iniciar (en emisor)

Para que dos o más tarjetas puedan comunicarse entre sí, ambas deben

estar sintonizadas exactamente en el mismo número de grupo: En el bloque `al iniciar`, colocad la orden `grupo de conjunto de radio 1` de la categoría "● Salida".

Inicializad la variable con la instrucción `definir mensaje en 0`.

Dibujad una antena con `mostrar leds` para avisar de que la micro:bit está preparada para enviar mensajes.

Paso 4: Indicar el código de mensaje a enviar (en emisor)

Ahora, configuraremos el botón A y el B para seleccionar el número del mensaje a transmitir:

Seleccionad en la categoría "● Entrada" el bloque `con el botón A pressed`.

Asignad un valor a transmitir con la orden `cambiar mensaje por -1`.

Para que muestre el número del mensaje, incluid el bloque `mostrar número 0`, cambiando el 0 por la variable `mensaje`.

Hacemos lo mismo para el botón B, debemos utilizar el bloque `cambiar mensaje por 1`.

Paso 5: Enviar el código de mensaje (en emisor)

Enviaremos el código del mensaje al pulsar los dos botones:

Seleccionad en la categoría "● Entrada" el bloque `con el botón A+B pressed`.

Encajad en su interior el bloque `enviar número por radio 1` de la categoría "● Salida". Sustituid el 1 por la variable `mensaje`.

Añadid el bloque mostrar icono con una flecha mirando hacia la derecha para señalar de forma visual que se ha emitido una señal. Haced una pausa de un segundo, y que vuelva a aparecer el icono de una antena.

Paso 6: Programar la recepción (en receptor)

Igual que hicimos en la otra tarjeta, en el bloque `al iniciar`, colocad la orden `grupo de conjunto de radio 1` de la categoría "● Salida". De esta manera, ambas tarjetas estarán sintonizadas.

Dibujad también una antena con `mostrar leds` para avisar de que la micro:bit está preparada para enviar mensajes.

Paso 7: Comprobar el mensaje recibido (en receptor)

En la categoría "● Entrada" seleccionad el evento `en el número recibido por radio valor`. En la variable, creada automáticamente, llamada "valor", se almacena el número de mensaje recibido.

Dentro del evento de recepción de radio, podéis colocar un condicional simple `si verdadero entonces`.

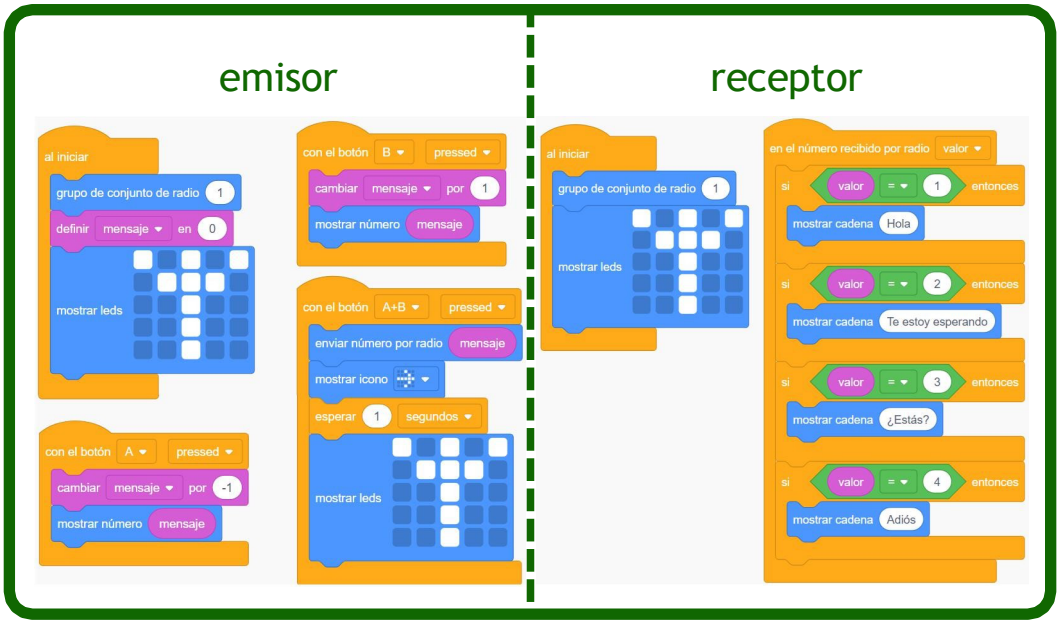
Reemplazad la casilla `verdadero` por el bloque `1 = 1` del bloque "● Matemáticas". Además, sustituid el primer 1 por la variable `valor`.

En su interior, mostrar un mensaje con el bloque `mostrar cadena`.

Podéis crear tantos `si verdadero entonces` como queráis, pero en cada uno de ellos debéis variar el número que tendría la variable `valor` y el mensaje a mostrar.

Paso 8: Simulación y comprobación en Tinkercad

Haced clic en "Iniciar simulación". Veréis que Tinkercad muestra las dos tarjetas micro:bit virtuales en la pantalla para simular la red. Con los botones A y B del emisor podréis reducir o aumentar el número de mensaje a enviar. Pulsadlos simultáneamente, para que ese código de mensaje se envíe. Veréis cómo la pantalla de esa placa muestra el icono de envío mientras que la pantalla de la segunda micro:bit recibe al instante la señal inalámbrica y muestra el mensaje correspondiente en su matriz de LED.



Repaso

- La sintonización del canal de comunicación permite aislar o conectar dispositivos asignándoles una frecuencia de trabajo compartida.
- Eventos como `en el número recibido por radio` son bloques especiales que no se ejecutan en bucle continuo, sino que actúan únicamente en el momento exacto en que captan una señal entrante en el canal establecido.

Desafío

¿Cómo podríais convertir este sistema de comunicación en un mando a distancia para enviar caritas de estado?

Intentad que la pulsación del botón A envíe el número 1 (que la otra placa interpretará dibujando una carita feliz) y la pulsación de la entrada B envíe el número 2 (que la otra placa interpretará como carita triste).

¿Qué ocurre si cambiáis el número de grupo en solo una de las dos micro:bit?

Proyecto 15: Lluvia de píxeles

Dificultad: ★★★☆☆ Tiempo: 40 m

Aprendizaje

Con este juego de recogida de gotas que caen, descubriremos cómo dibujar y mover puntos en la pantalla. Es importante saber que la pantalla funciona como un plano de coordenadas numéricas. La posición horizontal se mide en el eje X (de 0 en el borde izquierdo a 4 en el borde derecho) y la posición vertical en el eje Y (de 0 en el borde superior a 4 en el borde inferior). Así, la esquina superior izquierda es el punto (0,0), el centro de la pantalla es el (2,2) y la esquina inferior derecha es el (4,4).

Aprenderemos a utilizar el bloque de azar para que los objetos caigan desde posiciones impredecibles en la parte superior de la pantalla.

Programaremos la verificación de impacto comprobando en un mismo instante si las coordenadas del objeto que cae coinciden exactamente con la posición de nuestra cesta.

Realización

Paso 1: Crear las variables del juego

Para controlar la posición de nuestro jugador en la base, la posición de la gota que cae y la puntuación, crearemos cinco variables: "jugador_x", "gota_x", "gota_y", "velocidad" y "puntos".

Paso 2: Inicializar la partida al arrancar

Al encender la tarjeta, configuraremos la posición inicial de la cesta en el centro de la fila inferior y colocaremos los puntos a cero. En el bloque `al iniciar`, colocad la orden `definir jugador_x en 2` (el centro horizontal de la matriz de LEDs). Colocad la orden `definir puntos en 0`.

Poned el bloque `definir gota_x en 0`, sustituyendo el 0 por `seleccionar de forma aleatoria entre 0 y 4` de la categoría "● Matemáticas". Colocad la `orden definir gota_y en 0` (la fila superior de la pantalla).

Para poder ajustar la velocidad del juego, añadiremos el bloque `definir velocidad en 400`. Esta cantidad indica la pausa que queremos que haga el sistema tras cada avance de la gota. La podremos modificar si el juego va demasiado lento o rápido.

Paso 3: Mover el jugador

Programaremos los botones de la placa para desplazar la cesta por la fila inferior (Y = 4) sin salirnos de los márgenes de la pantalla:

Seleccionad en la categoría "● Entrada" el bloque `con el botón A pressed`.

Para evitar que el jugador se desplace por debajo del cero, añadid un condicional simple `si verdadero entonces`, sustituyendo `verdadero` por `jugador_x > 0`. En su interior, colocad la orden `cambiar jugador_x por -1`. De esta manera, cuando llegue a la posición 0 no sigue disminuyendo el valor de cero aunque sigamos pulsando el botón A.

Actuaremos de manera similar para desplazar al jugador hacia la derecha hasta su límite máximo. Seleccionad en la categoría "● Entrada" un bloque `con el botón B pressed`.

Añadid el condicional simple `si verdadero entonces` comprobando si `jugador_x < 4`. Dentro colocad la orden `cambiar jugador_x por 1`.

Paso 4: Animar la caída de la gota en el bucle principal

En el bloque continuo, haremos que la gota descienda paso a paso por la matriz de LED:

En el bucle `siempre`, colocad primero la orden `borrar pantalla`.

Encended el LED del jugador con el bloque `trazar x y` de la categoría "● Salida", asignando la variable `jugador_x` en la casilla X y el número 4 en la casilla Y.

Encended el LED de la gota con el bloque `trazar x y`, asignando la variable `gota_x` en la casilla X y la variable `gota_y` en la casilla Y.

Añadid el bloque `esperar 1 segundos`, pero cambiando el 1 por la variable `velocidad` y los segundos por milisegundos para marcar la velocidad de caída.

Incrementad la posición vertical de la gota con la orden `cambiar gota_y por 1`.

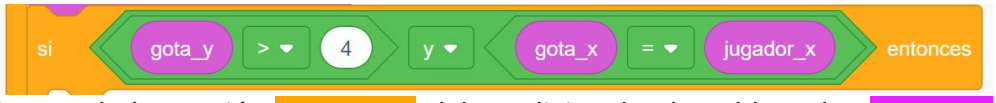
Paso 5: Comprobar captura o reinicio con el bloque verdadero y verdadero

Evaluaremos si la gota ha llegado al fondo y el jugador ha conseguido atraparla a tiempo. Dentro del bucle `siempre`, justo debajo del cambio de posición de la gota, colocad el bloque condicional `si verdadero entonces`.

Reemplazad la casilla `verdadero` por el bloque `verdadero y verdadero` de la categoría "● Matemáticas".

En la primera casilla de `verdadero y verdadero`, evaluad si `gota_y > 4`.

En la segunda casilla de `verdadero y verdadero`, evaluad si `gota_x = jugador_x`.



Dentro de la sección `entonces` del condicional colocad la orden `cambiar puntos por 1`. Si ha llegado ahí es porque hemos capturado la gota.

Justo a continuación, aún dentro de la sección `entonces`, haced reaparecer la gota arriba con `definir gota_x en seleccionar de forma aleatoria entre 0 y 4`, y con `definir gota_y en 0`, de la misma manera que ya pusimos en `al iniciar`.

Incluid un `mostrar número puntos` para ir conociendo la puntuación.

Ahora comprobaremos si la gota ha llegado al fondo y el jugador no la ha capturado. Dentro del bucle `siempre`, justo a continuación, colocad el bloque condicional `si verdadero entonces`.

Reemplazad la casilla `verdadero` por el bloque `verdadero y verdadero` de la categoría "● Matemáticas".

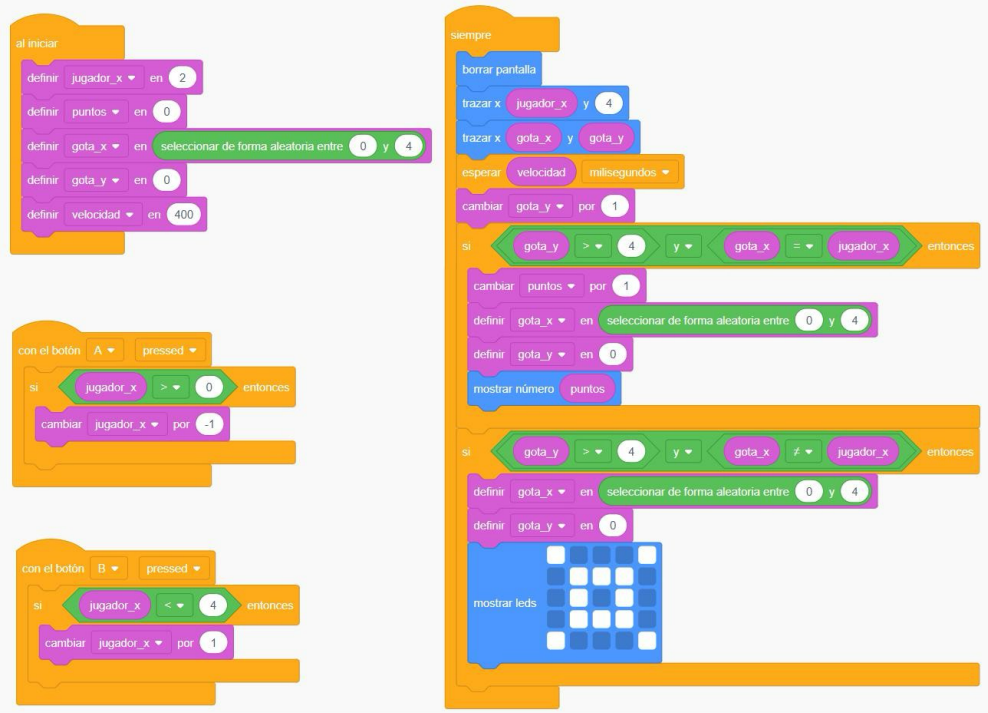
En la primera casilla de `verdadero y verdadero`, evaluad si `gota_y > 4`.

El segundo `verdadero` de `verdadero y verdadero`, cambiadlo por la evaluación `gota_x ≠ jugador_x` (fijaos que es ≠, indicando la no igualdad).

En su interior colocad los mismos bloques que en el condicional anterior, pero eliminando el que incrementaba la variable `puntos`. Además, ahora, en vez de mostrar la puntuación, que muestre un icono que confirme que no capturasteis la gota.

Paso 6: Simulación y comprobación en Tinkercad

Haced clic en "Iniciar simulación". Veréis que un punto de luz comienza a caer desde la parte superior de la matriz de LED. Pulsad el botón A para desplazar vuestro punto de la base hacia la izquierda y el botón B para moverlo a la derecha. Comprobad cómo al hacer coincidir vuestra posición con la trayectoria del punto que cae aumenta la puntuación y la gota se reinicia arriba.



Repaso

- La matriz de LED de la micro:bit se organiza como un plano cartesiano de 5x5 donde el punto (0,0) está en la esquina superior izquierda y el punto (4,4) en la esquina inferior derecha.
- La generación de números aleatorios añade dinamismo a los juegos, evitando que los elementos caigan siempre por el mismo sitio.

Desafío

¿Os acordáis de la variable `velocidad`? ¿Cómo podríais conseguir que el juego sea cada vez más difícil a medida que atrapáis más gotas?

Intentad que cada vez que la condición de captura sea verdadera, la variable `velocidad` se reduzca en 10 milisegundos.

¿Y si quisiera que el juego fuera más difícil tanto si se captura la gota como si no se captura?

Reto 1: Generador de números de lotería

Dificultad: ★★☆☆☆ **Tiempo:** 30 m

Enunciado

Hemos llegado a un punto muy importante. Tras haber completado 15 proyectos guiados, ya domináis el uso de los botones, los sensores de movimiento y orientación, el manejo de variables, los bucles y la generación de números aleatorios en la tarjeta micro:bit. A partir de este momento, vais a asumir el papel de desarrolladores independientes: yo os plantearé el reto y las condiciones que debe cumplir el dispositivo, y seréis vosotros quienes debáis diseñar la lógica de bloques, probar la solución en el simulador y corregir los fallos que surjan.

El objetivo de este primer desafío es transformar la tarjeta en un generador automático de números de lotería (o bombo digital de sorteo) que sea capaz de elegir y mostrar cifras al azar dentro de un rango determinado cada vez que el usuario lo solicite.

Para dar por resuelto el proyecto, vuestro programa en Tinkercad debe cumplir estrictamente con los siguientes requisitos de funcionamiento:

- Al iniciar, el sistema debe indicar de forma clara que está listo para funcionar.
- Al pulsar el botón A, debéis programar un pequeño efecto de espera antes de obtener y mostrar el resultado.
- La cifra elegida debe generarse mediante la instrucción `seleccionar de forma aleatoria entre 1 y 10` (ajustando, por supuesto, los límites numéricos para que elija una cifra comprendida entre 1 y 49).
- Una vez montada vuestra estructura lógica, debéis hacer clic en "Iniciar simulación" para comprobar que funciona correctamente.

Reto 2: Alarma anticongelación

Dificultad: ★★☆☆☆ Tiempo: 30 m

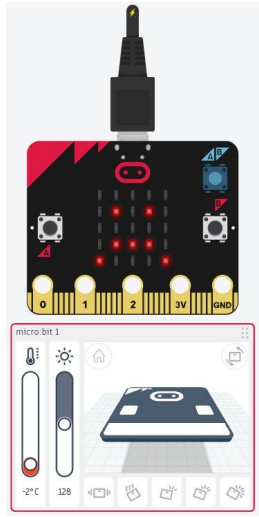
Enunciado

Las heladas nocturnas pueden causar graves daños en los cultivos y congelar las tuberías de agua en las viviendas. Para prevenir estos problemas, los agricultores y técnicos utilizan sistemas de monitoreo automático que lanzan una alerta visual o sonora cuando la temperatura cae a niveles peligrosos, permitiéndoles tomar medidas a tiempo.

Vuestro objetivo es diseñar un simulador en Tinkercad utilizando la tarjeta micro:bit que actúe como un sensor de prevención de congelación. El programa debe realizar las siguientes tareas de forma continua:

- Medir continuamente la temperatura ambiente utilizando el sensor integrado de la micro:bit.
- Si la temperatura es inferior o igual a 0 °C, el sistema debe activar la alarma: Mostrar en la pantalla LED una carita triste o un icono de advertencia.
- Si la temperatura es superior a 0 °C, la pantalla LED mostrará una carita feliz o un icono de estado correcto.

Necesitarás un bucle principal `para siempre` junto con un bloque condicional `si verdadero entonces ... si no ...`. Piensa bien qué condición matemática utilizar para comparar la temperatura. Si decidís utilizar una variable, llamadla `temperatura`.



Reto 3: Reloj de arena digital

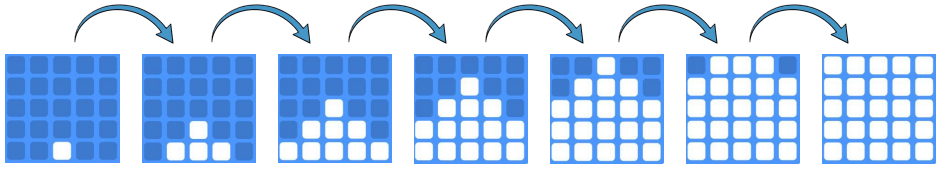
Dificultad: ★☆☆☆☆ **Tiempo:** 25 m

Enunciado



Los relojes de arena se han utilizado durante siglos para medir intervalos de tiempo mediante la caída de finos granos de arena. En la era digital, podemos simular esta misma idea utilizando la matriz de luces LED de la micro:bit. En este proyecto crearéis un temporizador visual que muestre de forma animada cómo la "arena" se vacía progresivamente en la pantalla antes de avisar al usuario de que el tiempo se ha agotado.

- Al encender la micro:bit, la pantalla LED debe mostrar un icono que indique que está preparada.
- El programa debe ir encendiendo luces de la pantalla LED como si fuera la parte inferior de un reloj de arena donde caen los granos de arena.



- Una vez que toda la "arena" haya caído al fondo de la pantalla, el sistema debe indicar que la cuenta atrás ha finalizado.

Debéis crear las imágenes de la pantalla cuadro a cuadro usando el bloque `mostrar leds` e ir introduciendo pausas con el bloque `esperar` entre cada cambio de dibujo para simular la velocidad del reloj.

Reto 4: Despertador de sol

Dificultad: ★★☆☆☆ Tiempo: 30 m

Enunciado

Algunos despertadores modernos no nos despiertan con un ruido molesto repentino, sino que simulan el amanecer encendiendo gradualmente una luz o detectando cuándo entra el primer rayo de sol por la ventana. De esta manera, nuestro cuerpo se despierta de forma natural. En este proyecto vais a programar la micro:bit para que funcione como un sistema de aviso que detecte cuando amanece en vuestra habitación utilizando exclusivamente los sensores propios de la tarjeta.

El programa en bloques debe realizar las siguientes tareas:

- Al iniciar el programa, la pantalla LED de la micro:bit debe estar totalmente apagada mientras haya oscuridad en la habitación.
- El sistema medirá constantemente la intensidad de luz del entorno mediante el sensor de luz integrado en la matriz de LEDs de la propia micro:bit.
- Según se vayan superando distintos umbrales de nivel de luz, la pantalla debe mostrar una imagen distinta que represente la salida del sol, muy similar a alguno de los iconos del reto anterior.

Utilizad un bucle `siempre` junto con varias estructuras condicionales `si verdadero entonces` para comprobar de forma continua si la luz leída supera los límites que has fijado.

Recordad que el bloque `nivel de luz` de la sección "● Entrada" utiliza la matriz de LEDs de la tarjeta para medir la luz ambiental.

Al ejecutar la simulación, utilizad el deslizador con el icono del sol para simular los distintos niveles de luz y comprobar si vuestro programa funciona.

Reto 5: Nivel digital

Dificultad: ★★☆☆☆ Tiempo: 30 m

Enunciado



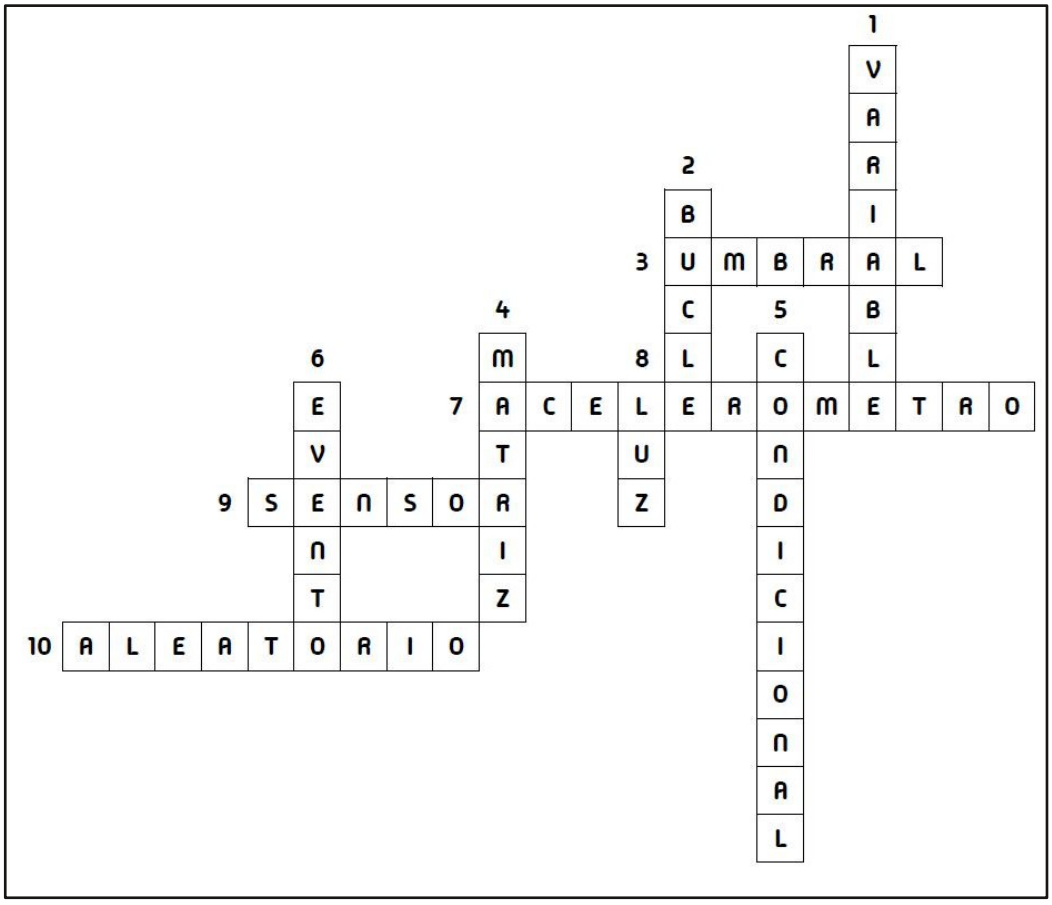
Cuando queremos colgar un cuadro, instalar un estante o comprobar si una mesa está bien nivelada, utilizamos un nivel de burbuja tradicional. Este instrumento contiene un tubo con líquido y una burbuja de aire que se desplaza hacia el centro únicamente cuando la superficie está completamente horizontal. En este reto vais a convertir tu tarjeta micro:bit en una herramienta de bricolaje digital capaz de detectar la inclinación de cualquier objeto.

- Al iniciar el programa, debe hacer una pequeña animación que indique que se está preparando.
- Luego, de forma continua, debe comprobar la posición de la tarjeta utilizando su acelerómetro (detectando la inclinación hacia la izquierda o hacia la derecha). Si la tarjeta se inclina hacia la izquierda, la matriz LED debe mostrar una flecha señalando a la derecha; por el contrario, si la inclinación es hacia la derecha, la flecha señalará a la izquierda. Cuando la tarjeta esté totalmente horizontal (con un margen de tolerancia cercano a los 0°), mostrará una flecha hacia arriba. Utiliza el bloque `rotación (°) roll` de la sección "● Entrada".

Necesitaréis emplear varios bloques condicionales `si verdadero entonces` para comparar el valor del ángulo de inclinación:

- Si la inclinación es menor que -10º, está inclinada a la izquierda.
- Si la inclinación es mayor que 10º, está inclinada a la derecha.
- Entre -10º y 10º, está nivelada.

Gimnasia mental: Soluciones

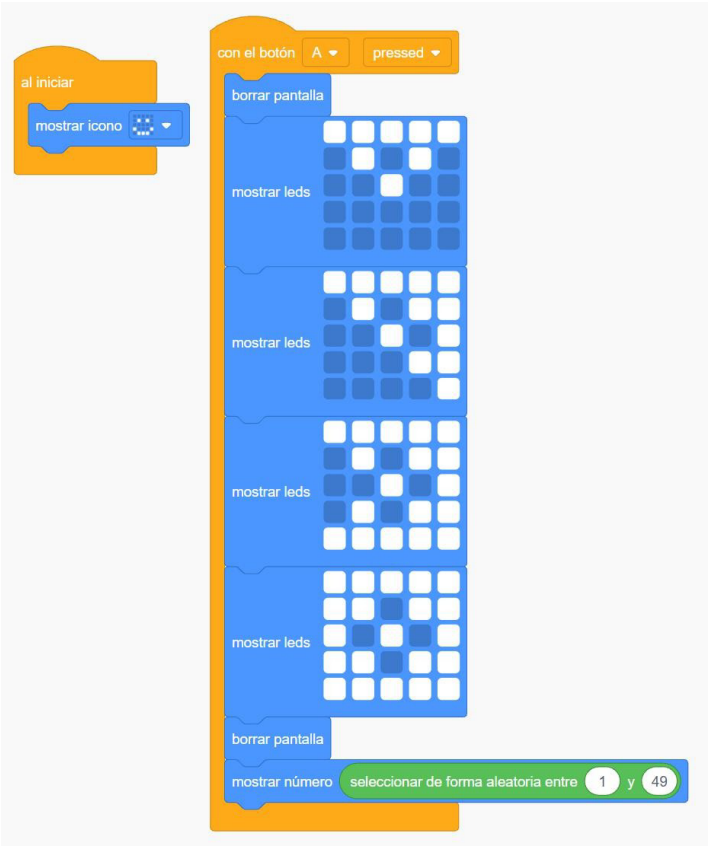


E	N	C	K	T	I	D	A	R
D	R	I	E	A	N	K	T	C
T	K	A	D	R	C	E	I	N
A	I	D	C	N	E	R	K	T
R	T	N	A	D	K	I	C	E
C	E	K	R	I	T	N	D	A
I	D	T	N	E	A	C	R	K
K	A	E	I	C	R	T	N	D
N	C	R	T	K	D	A	E	I

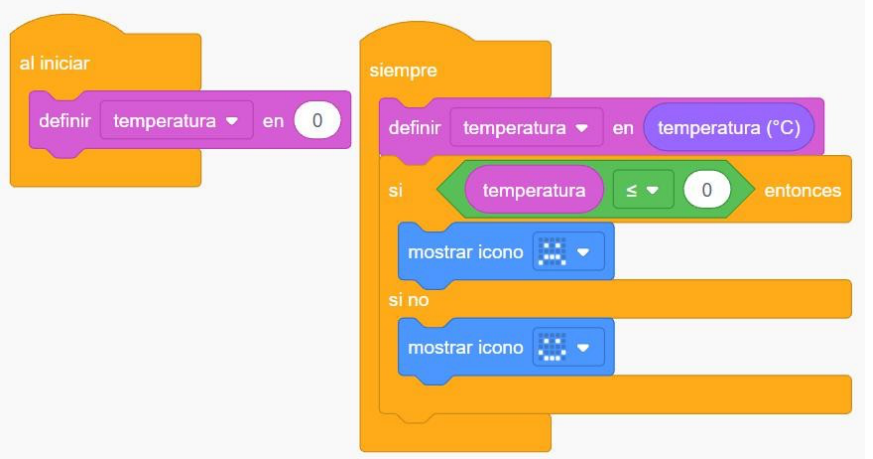


Reto: Soluciones

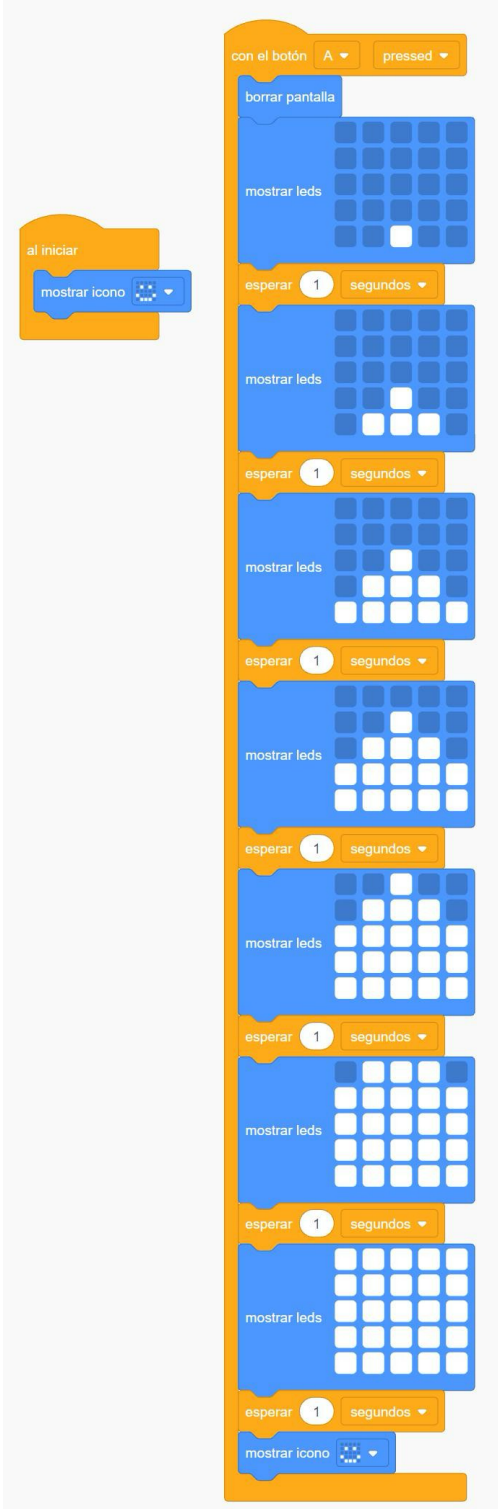
Reto 1



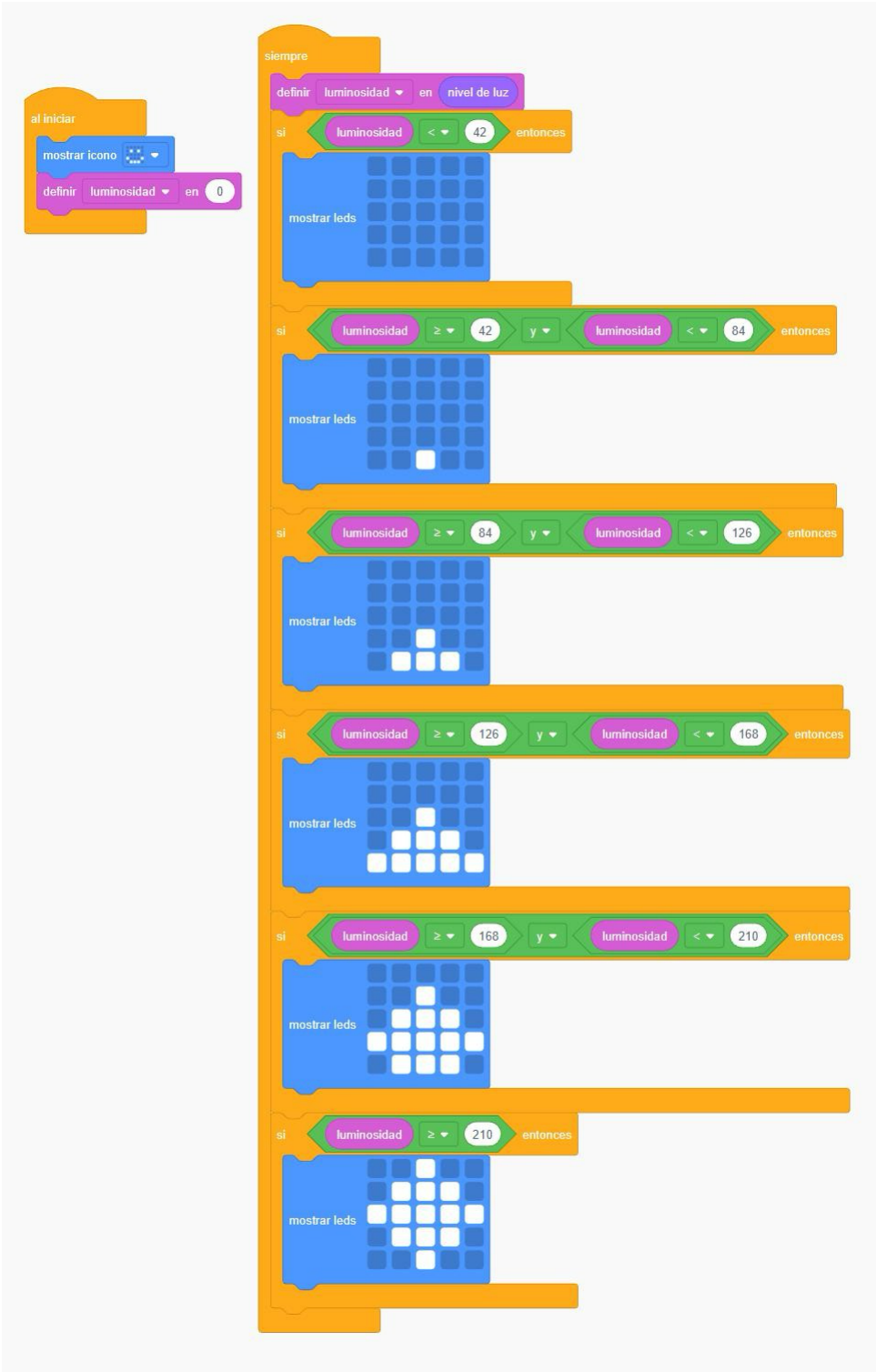
Reto 2



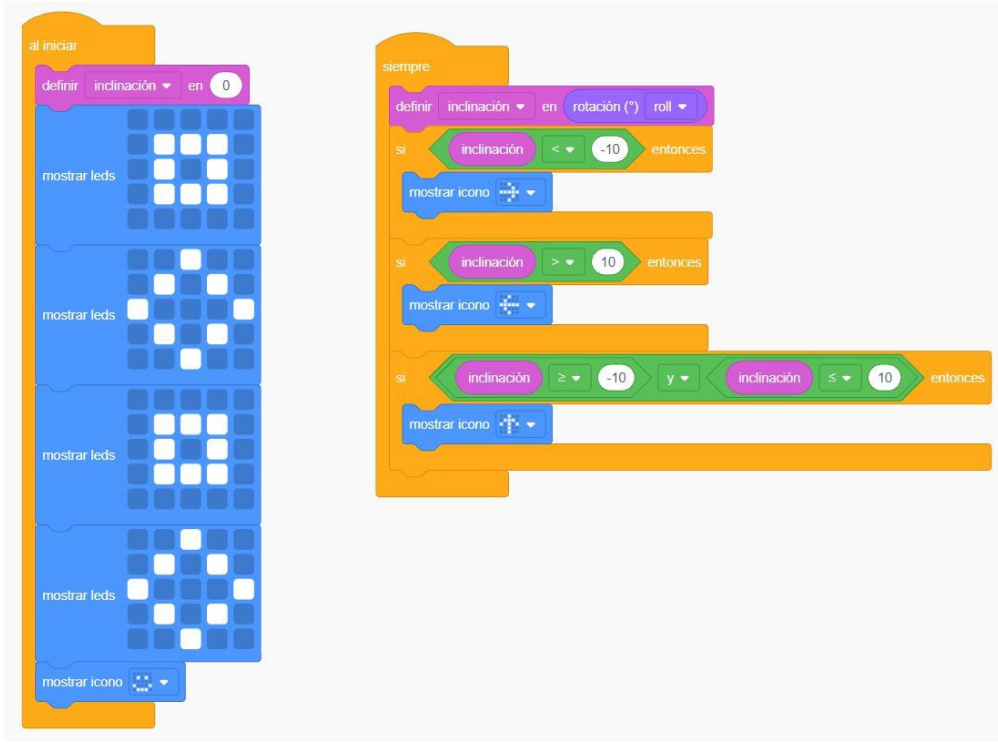
Reto 3


al iniciar
mostrar icono
con el botón A pressed
borrar pantalla
mostrar leds
esperar 1 segundos
mostrar leds
esperar 1 segundos
mostrar leds
esperar 1 segundos
mostrar leds
esperar 1 segundos
mostrar leds
esperar 1 segundos
mostrar leds
esperar 1 segundos
mostrar leds
esperar 1 segundos
mostrar icono

Reto 4

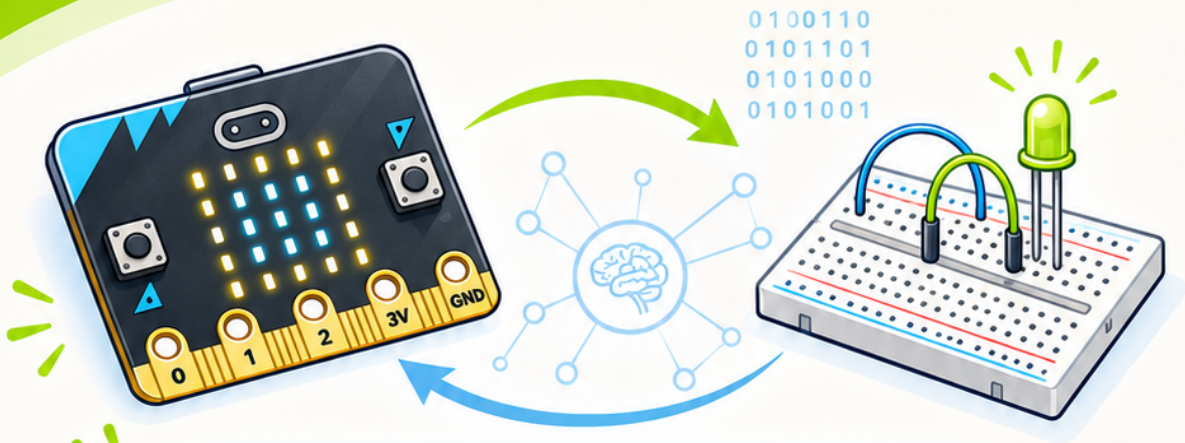

al iniciar
mostrar icono
definir luminosidad en 0
siempre
definir luminosidad en nivel de luz
si luminosidad < 42 entonces
mostrar leds
si luminosidad ≥ 42 y luminosidad < 84 entonces
mostrar leds
si luminosidad ≥ 84 y luminosidad < 126 entonces
mostrar leds
si luminosidad ≥ 126 y luminosidad < 168 entonces
mostrar leds
si luminosidad ≥ 168 y luminosidad < 210 entonces
mostrar leds
si luminosidad ≥ 210 entonces
mostrar leds

Reto 5


al iniciar
definir inclinación en 0
mostrar leds
mostrar leds
mostrar leds
mostrar leds
mostrar icono
siempre
definir inclinación en rotación (°) roll
si inclinación < -10 entonces
mostrar icono
si inclinación > 10 entonces
mostrar icono
si inclinación ≥ -10 y inclinación ≤ 10 entonces
mostrar icono

© José Antonio Pascual del Coso, 2026

© Palmito Books, S.L., 2026
Calle Pedro García Villalba, 79, 2ºC
30150 La Alberca, Murcia
ESPAÑA



Generación IA: 20 proyectos micro:bit. Nivel: iniciación es una guía práctica para dar los primeros pasos en programación, pensamiento computacional y creación tecnológica con micro:bit. A través de 20 proyectos sencillos y motivadores, el lector aprenderá a utilizar las funciones básicas de la placa, comprenderá cómo llevar una idea del entorno virtual a la práctica y desarrollará habilidades para experimentar, resolver problemas y crear soluciones propias. La obra está pensada para alumnado y personas que se inician desde cero, con un enfoque progresivo, accesible y orientado al aprendizaje activo mediante la exploración, la simulación y la creatividad.



0100110
0101101
0101000

0100011
0101100
0101001

PALMITO BOOKS

ISBN 979-13-88535-26-0