# FICHA 3 EXPEDICIÓN ESPELEOLÓGICA

*Coding for kids, un programa del* ***British Council-MinTIC-CpE***

## Aprendizajes

Al final de esta actividad se espera que puedas:

- Utilizar variables booleanas de entrada que simulan la acción de sensores.
- Comunicar instrucciones utilizando la pantalla de LED y un código de flechas.
- Interpretar una secuencia de instrucciones para resolver un problema como el de un laberinto.
- Interpretar un diagrama de flujo para resolver problemas como el de un laberinto.
- Utilizar operaciones lógicas para decidir qué acción se ejecuta.
- Utilizar lazos que se repiten hasta terminar la tarea.

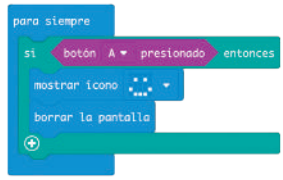

## Sesión 1

### Lo que sabemos, lo que debemos saber

En las fichas anteriores ya has trabajado ***entradas booleanas*** (los botones) y la salida de LED. Igualmente, has utilizado bloques que representan algunas acciones o instrucciones que se deben repetir.

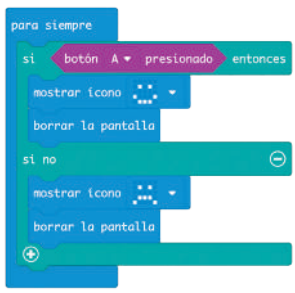
Las ***variables booleanas*** pueden asumir dos valores solamente: ***verdadero*** o ***falso.***

Cuando el ***Botón A está oprimido,*** su valor es ***verdadero*** y cuando no, es ***falso.*** En este ejemplo, si presionas ***A*** verás una cara feliz.
Esta es una nueva forma de controlar la realización o no de ciertas instrucciones.


En este caso se ejecutará la instrucción mostrar cara feliz si A está presionado. ¿Con qué objetivo se coloca borrar pantalla? ¿Qué sucede si no se coloca esta instrucción? Prueba sobre el editor tu predicción.

Ahora mira el diagrama de flujo que aparece a la izquierda y la programación en bloques que se visualiza en la parte de abajo::




En este caso, al no estar oprimido el botón A, se verá una ***cara triste.*** En caso contrario, si lo oprimes, verás una ***cara feliz.*** De hecho, esta estructura se puede complicar aún más si se oprime el símbolo + (más), quedando el bloque como se muestra (si no).

Si se oprime el símbolo + con el ratón, la estructura permite anidar de nuevo otra condición.
¿Se requiere en este caso colocar borrar pantalla? ¿Por qué sí o por qué no? Verifica en el editor.

**¿Está oprimido el botón A?**

↓ **Sí**

**¿Está oprimido el botón B?**

↓ **Sí**

**¿La temperatura es mayor de 25°?**

↓ **Sí**

**Acción si las tres condiciones son verdaderas**

En algunos casos se tienen dos o tres condiciones seguidas, como en el diagrama de la izquierda. En estos casos se pueden reemplazar varias condiciones por una sola verificación, que incluya las dos o tres condiciones:

¿Está oprimido el botón A y está oprimido el botón B y la temperatura es mayor de 25°?

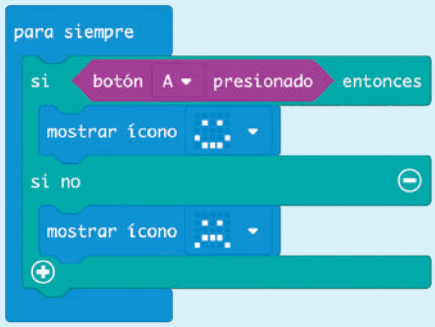
↓ Sí

**Acción si TODAS las tres condiciones son verdaderas**

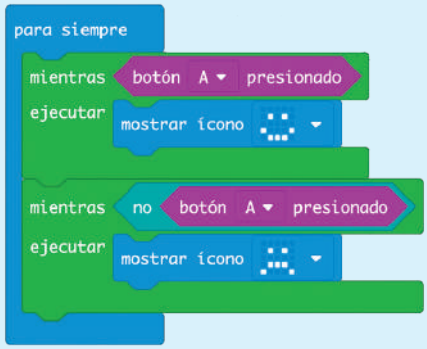
Lo mismo puede ser expresado en la condición de bloques por el conjunto de bloques que aparece abajo. Para que salga la cara feliz se requiere que las tres condiciones sean verdaderas. Si alguna es falsa, no debe mostrarse la cara feliz:


**RESUMEN**

Las instrucciones ***si*** (condición) ***entonces*** (instrucciones) ***si no entonces*** (instrucciones) se representa por:

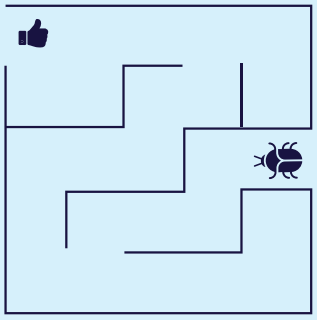



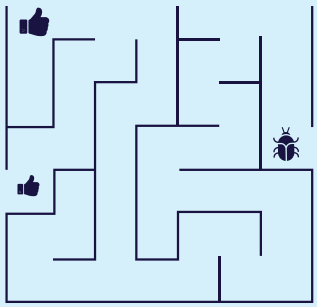

Compara este diagrama de bloques con el que está a continuación, ¿hacen lo mismo?




Recuerda que puedes acceder al editor Makecode en

***makecode.microbit.org***





## Desconectadas

Piensa en una caverna que tiene una entrada y una salida. Se sabe también que es un laberinto de túneles formados por el paso del agua a lo largo de muchos años. Un grupo de espeleólogos, en el cual tú estás, debe atravesar la caverna completa. Tú conoces el algoritmo de "**seguir la pared**", que se encuentra abajo a la izquierda, y propones que sea utilizado. Pero antes de entrar a la caverna te piden que demuestres que el algoritmo funciona bien. Para ello te dan los dos laberintos que se muestran a la izquierda. Con una copia de los laberintos con cinta en el piso o en papel y una ficha, verifica que el algoritmo funciona bien. Si cuentas con otras personas, asigna los siguientes roles:

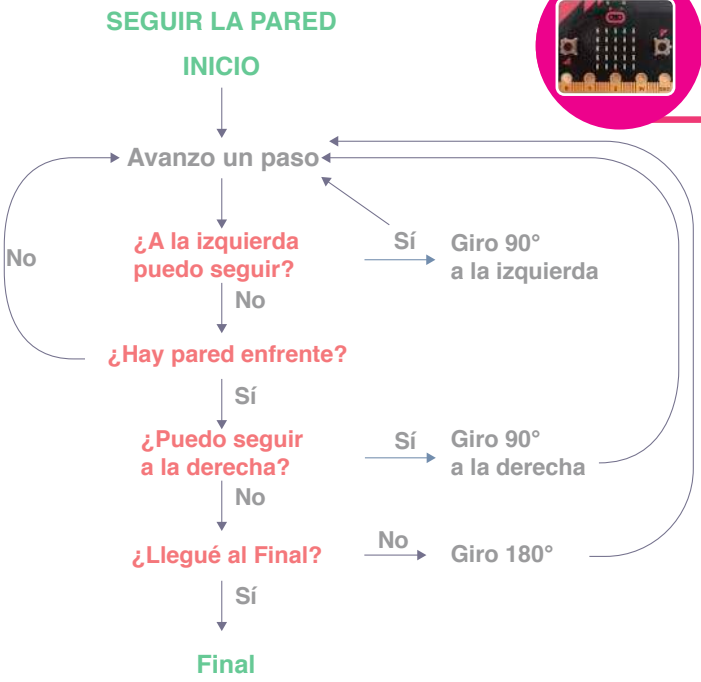
•**Depurador**: sigue el algoritmo colocando una ficha en la instrucción que se está ejecutando del diagrama de flujo e indicando en voz alta la instrucción que corresponde.

•***Procesador:*** pone un objeto, representado por una ficha, en la entrada del laberinto y lo mueve, o se desplaza en el laberinto hecho en el piso. En el caso del objeto, debe tener claro cuál es el frente de la ficha al irla desplazando.

•***Medidor de complejidad:*** va contando los pasos requeridos para salir del laberinto.

•***Verificador:*** si hay alguien más en el grupo, esta persona debe verificar que se siga la secuencia de instrucciones y cuenta cuántos pasos se dan.

**Al pasar al segundo laberinto, si hay más personas, cambia los roles.**
Terminada la labor, compara el número de pasos dados en los dos laberintos. También podrás buscar otros laberintos más complejos para probar el algoritmo y verificar qué tantos pasos debes dar para resolver el problema.






***Si tienes una micro:bit a tu alcance, es el momento de hacer un programa que te ayude a salir de un laberinto. Este programa es un reto.***

Una flecha adelante te pregunta si hay pared adelante. Una flecha a la izquierda te pregunta si hay muro a la izquierda, una flecha a la derecha te pregunta si hay muro a la derecha. ¿Qué tal así?

Tú harás de sensor y contestarás **sí** oprimiendo el botón A y **no** oprimiendo el botón B.

Estas son entradas, algo que hay que "leer" del medio exterior. Igualmente hay una salida: avanzar/no avanzar.

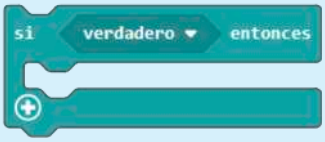
Encuentra cómo te dirá la ***micro:bit*** que avances o gires.

Si tienes el editor en una APP de un celular, ¿podrás hacer el mismo ejercicio?

## Traduciendo a la micro:bit

Recuerda que:
Todo programa en la ***micro:bit*** debe estar en un bucle de repetir general como el siguiente (hay otros):


Los bloques condicionales encajan perfectamente en los bloques de bucle. El hexágono que está al lado de la palabra clave “si” representa una variable ***booleana,*** por lo tanto, asume dos posibles valores: verdadero o falso.
Los bloques que insertes en el bloque condicional “si”, solo se ejecutarán cuando el valor del hexágono sea “verdadero”.




El bloque condicional “si, si no” tiene un espacio adicional para agregar bloques que se ejecutarán cuando el valor del rombo sea “falso”. Puedes convertir el bloque "si" en uno "si, si no" presionando el signo “+” de la parte inferior.


## Sesión 2

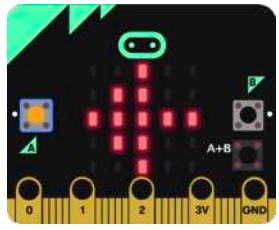
### Conectadas: manos a la Micro:bit

Es hora de seguir profundizando tus conocimientos sobre la ***micro:bit***. Hasta ahora has explorado el entorno de programación, el simulador y los bloques para crear bucles. En esta ocasión, revisarás los bloques de lógica condicional. Recuerda que para trabajar con la ***micro:bit*** necesitarás entrar a **Makecode** en tu computador o al editor en línea si tienes acceso a Internet.
Para poner a prueba los nuevos bloques aprendidos, crearás un dispositivo que te permita ayudar a las personas que se desplazan en bicicleta a transitar de forma más segura por las vías permitidas.
El dispositivo permitirá tener luces informativas mientras se monta en bicicleta. Es posible extender los botones de la ***micro:bit*** para que sean presionados desde los manubrios, donde se colcan las manos. Sin embargo, con fines ilustrativos y a modo de prueba de concepto de tu diseño, usarás los botones A y B que ya conoces.

El dispositivo funcionará de la siguiente manera:

**a.** Cuando se presione el botón A, la ***micro:bit*** deberá mostrar una luz direccional a la izquierda parpadeando.
**b.** Cuando presione el botón B, la ***micro:bit*** deberá mostrar una luz direccional a la derecha parpadeando.
**c.** Cuando se presionen los dos botones, la **micro:bit** deberá mostrar un indicativo de parada. ¿Cuál te parece más apropiado?

Es hora de programar y simular en el editor ***Makecode*** el programa para verificar su funcionamiento.

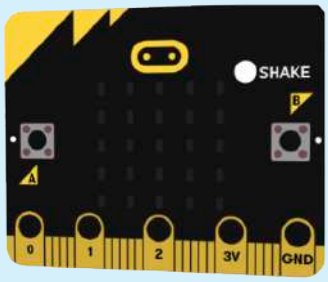



**1.** ¿A qué parte del funcionamiento corresponde el programa que se muestra a la derecha?

**2.** Al programarlo en el editor ***Makecode*** ¿qué sucede si retiramos el bloque “borrar la pantalla”?

**3.** Ahora debes programar el resto del funcionamiento. En este mismo bloque “para siempre”, agrega las condiciones restantes.

**4.** Cuando tu código incluye la condición de A+B, en el simulador aparece un tercer botón para probar tu código.





## *Trabajando con otras entradas*

Ya conoces el sensor de temperatura, recuerda que es otra variable de entrada. ¡La ***micro:bit*** posee más sensores! En esta ocasión usarás el sensor llamado ***acelerómetro.*** Este sensor mide de cierta forma el movimiento del dispositivo.

Imagina que tienes una botella llena de agua y que al taparla queda una burbuja de aire atrapada en su interior. A medida que cambias de posición la botella, la burbuja se desplaza para quedar siempre lo más arriba posible. La burbuja se mueve tan rápido como muevas la botella. Así, el acelerómetro también puede saber en qué posición se encuentra la **micro:bit:** logotipo arriba, logotipo abajo, inclinado, etc.

También puede saber si se agita el dispositivo y qué tan rápido se hace: 3 g, 6 g, 8 g. Esta agitación se mide con respecto a la gravedad, así como cuando un vehículo acelera y sientes que algo te presiona contra el asiento, 3 g significa una presión equivalente al triple de la gravedad.

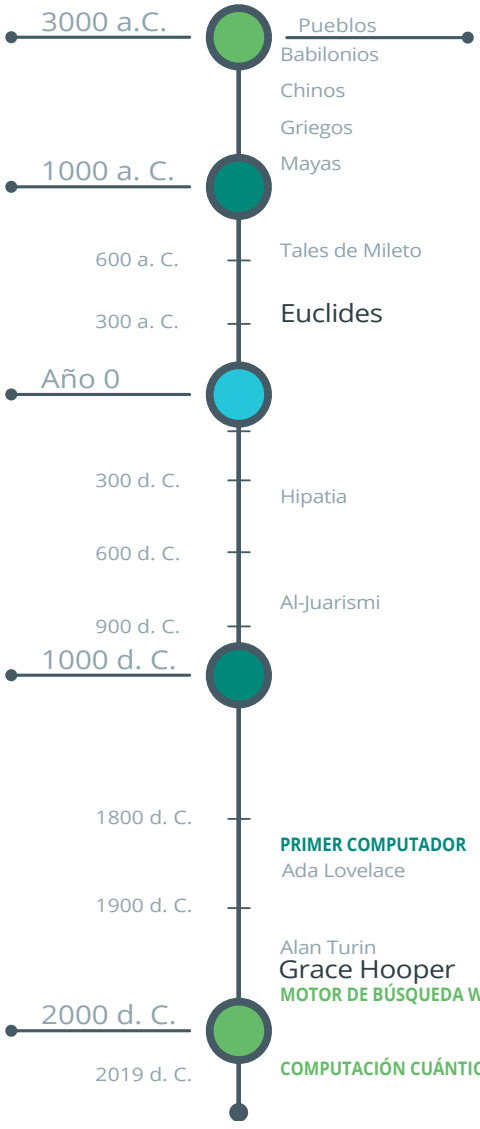
Puedes usar el acelerómetro para hacerte más visible mientras estés montando bici. Teniendo en cuenta que el dispositivo se estará agitando a medida que te mueves, podrías mostrar flechas que indiquen que te estás desplazando hacia adelante.

### *Aplicando lo aprendido*

**Desplazamiento hacia adelante**

1. El bucle presentado en el diagrama de la izquierda, es un bloque general como el bucle “para siempre”. Dentro de este bucle hay dos bloques que se repetirán uno tras otro hasta que se termine la tarea.
2. Antes de implementar el diagrama mencionado, intenta predecir lo que ocurrirá al ejecutarlo. Recuerda que el bloque “para siempre” se está ejecutando en todo momento. En este caso la entrada “agitado” del acelerómetro, es una variable booleana que puedes identificar por la forma del bloque.
3. Al agregar el sensor acelerómetro, el simulador presenta un botón que dice “SHAKE” (agitar) para representar que la tarjeta está siendo agitada.
4. ¿Ves la flecha desplazarse? ¿Podrías mejorar este desplazamiento? Considera que son como dos fotos que se muestran una tras otra y dan la sensación de movimiento; podrías agregar fotos intermedias para hacer el movimiento más fluido.
5. Complementa nuestro sistema de luces agregando animaciones para todos los indicadores.

## Un poco de historia




### Euclides (325 a. C. - 265 a. C.)

Matemático y geómetra griego.

Se le reconoce como el fundador de la geometría. Vivió en Alejandría, hoy norte de Egipto. En su escrito "Elementos de Euclides", se recopila todo el conocimiento de la época de manera formal y a partir de tan solo cinco postulados. En este documento también se presentó un algoritmo para calcular el máximo común divisor de dos números. Dicho algoritmo de más de 2000 años de antigüedad es conocido como el algoritmo de Euclides, y es incluso usado en la actualidad.

### Grace Hooper (1906 d. C. - 1992 d. C.)

Científica de la computación estadounidense.

Fue la primera persona en programar el primer computador electromecánico de la historia: el MARK I, basado en la máquina analítica de Charles Babbage y construido por IBM a petición de Howard Aiken, estudiante de la Universidad de Harvard. Entre 1950 y 1970 desarrolló el primer compilador (traductor a lenguaje de la máquina) para un lenguaje de programación, así como también propició métodos de depuración y validación para programas.


## Para ir más lejos

El programa que creaste en la sección anterior puede ser muy útil cuando montes en bici para indicar a quienes estén a tu alrededor si vas a girar a la izquierda o a la derecha, si estás en movimiento o si vas a parar. Haciendo algunos cambios puedes hacer el programa aún mejor. Has notado que al montar en bicicleta te inclinas ligeramente hacia el mismo lado hacia el que estás girando? ¿Cómo crees que puedes usar este fenómeno para mejorar tu programa? Como se mencionó anteriormente en esta ficha, la ***micro:bit*** cuenta con un acelerómetro. Además de medir si hay cambios en el movimiento, el acelerómetro puede indicar si la **micro:bit** está completamente horizontal o inclinada hacia la izquierda o hacia la derecha. Si fijas la ***micro:bit*** a tu cuerpo no necesitarías presionar los botones A o B para indicar un giro, la ***micro:bit*** podría usar su acelerómetro para determinar si estás girando y en qué dirección. Usa el bloque que se encuentra en el menú “Entrada” que se muestra a continuación:


## Lo que hemos aprendido

Revisa y completa la siguiente tabla marcando una X en la columna que mejor represente tu aprendizaje:

| Verifica los aprendizajes logrados | Sí | Algo | No |
|---|---|---|---|
| Utilizar variables booleanas de entrada que simulan la acción de sensores. | | | |
| Comunicar instrucciones utilizando la pantalla de LED y un código de flechas. | | | |
| Interpretar una secuencia de instrucciones para resolver un problema como el de un laberinto. | | | |
| Interpretar un diagrama de flujo para resolver problemas como el de un laberinto. | | | |
| Utilizar operaciones lógicas para decidir qué acción se ejecuta. | | | |
| Utilizar lazos que se repiten hasta terminar la tarea. | | | |

Selecciona la opción que mejor represente tu opinión:

| Contesta las siguientes preguntas | Sí | Algo | No |
|---|---|---|---|
| Las actividades realizadas fueron difíciles. | | | |
| Las actividades me motivaron. | | | |
| Siento que aprendí muchas cosas. | | | |
| Aún me quedan muchas dudas sobre lo que hice. | | | |