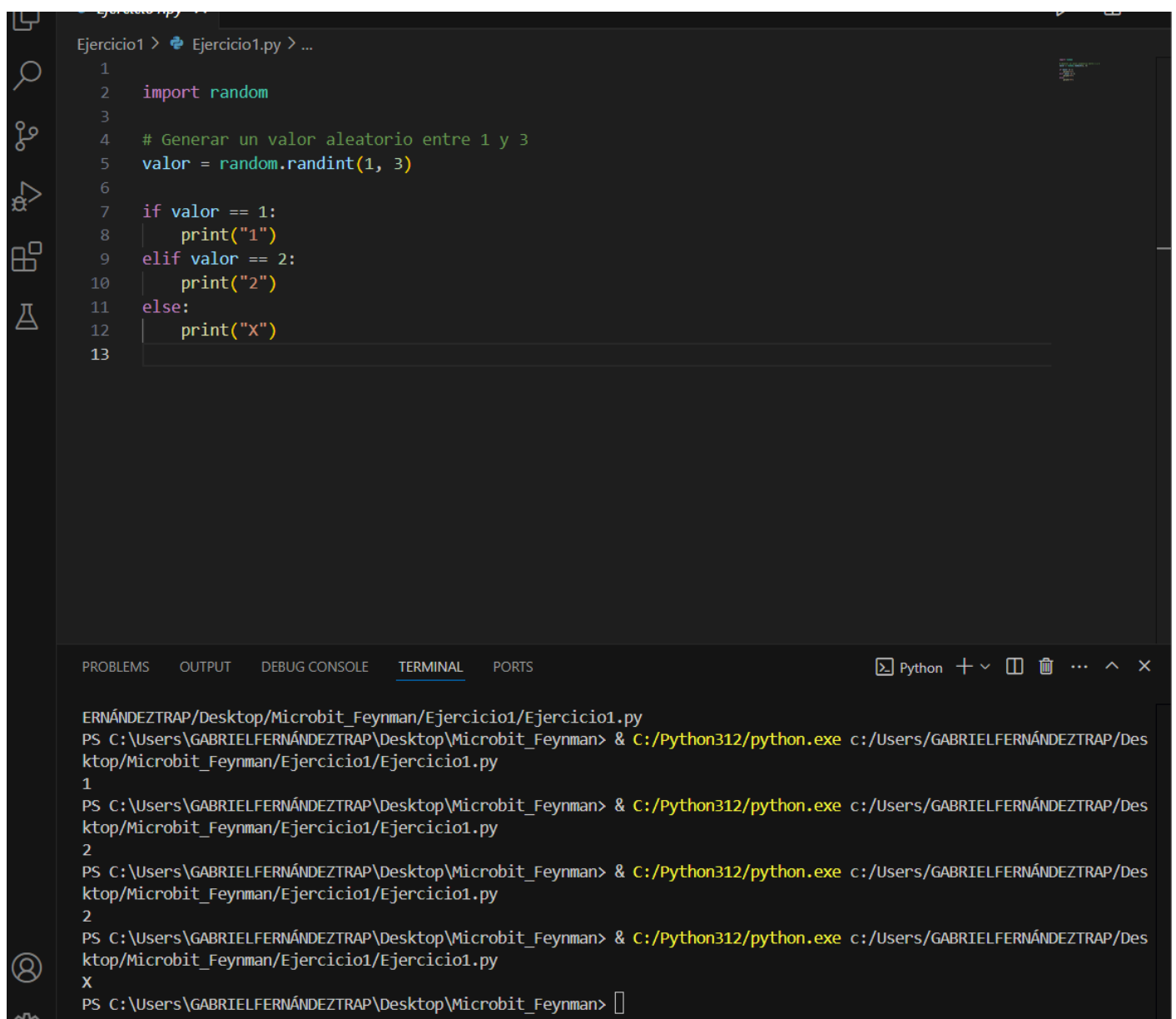


# Ejercicios Feynman-Microbit

**Ejercicio 1:** ¿Para qué sirve el siguiente programa? ¿Qué pasa si el valor es 3?

```
import random
valor = random.randint(1, 3)

if valor == 1:
    print("1")
elif valor == 2:
    print("2")
else:
    print("X")
```



The screenshot shows a Python IDE with a dark theme. The editor window displays the code from the previous block. The terminal window at the bottom shows the execution of the program three times. The first two runs output '1' and '2' respectively, while the third run outputs 'X'.

```
Ejercicio1 > Ejercicio1.py > ...
1
2 import random
3
4 # Generar un valor aleatorio entre 1 y 3
5 valor = random.randint(1, 3)
6
7 if valor == 1:
8     print("1")
9 elif valor == 2:
10    print("2")
11 else:
12    print("X")
13
```

PROBLEMS OUTPUT DEBUG CONSOLE **TERMINAL** PORTS Python + - [ ] [ ] ... ^ x

```
ERNÁNDEZTRAP/Desktop/Microbit_Feynman/Ejercicio1/Ejercicio1.py
PS C:\Users\GABRIELFERNÁNDEZTRAP\Desktop\Microbit_Feynman> & C:/Python312/python.exe c:/Users/GABRIELFERNÁNDEZTRAP/Desktop/Microbit_Feynman/Ejercicio1/Ejercicio1.py
1
PS C:\Users\GABRIELFERNÁNDEZTRAP\Desktop\Microbit_Feynman> & C:/Python312/python.exe c:/Users/GABRIELFERNÁNDEZTRAP/Desktop/Microbit_Feynman/Ejercicio1/Ejercicio1.py
2
PS C:\Users\GABRIELFERNÁNDEZTRAP\Desktop\Microbit_Feynman> & C:/Python312/python.exe c:/Users/GABRIELFERNÁNDEZTRAP/Desktop/Microbit_Feynman/Ejercicio1/Ejercicio1.py
X
PS C:\Users\GABRIELFERNÁNDEZTRAP\Desktop\Microbit_Feynman> [ ]
```

Este programa sirve para **generar un número aleatorio entre 1 y 3** y luego **imprimir un resultado dependiendo del valor generado**. Vamos a desglosarlo:

---

## ¿Cómo funciona?

Importa la librería `random`:

```
import random
```

1. Esto permite generar números aleatorios en el programa.

Genera un número aleatorio:

```
valor = random.randint(1, 3)
```

2.
  - o `random.randint(1, 3)` genera un número entero aleatorio entre 1 y 3, incluyendo ambos límites.

Condicionales:

```
if valor == 1:
    print("1")
elif valor == 2:
    print("2")
else:
    print("X")
```

3.
  - o Si el número aleatorio (`valor`) es:
    - 1, imprime 1.
    - 2, imprime 2.
    - 3, imprime X (cualquier otro valor fuera de 1 o 2 entrará en este caso, pero el rango ya está limitado a 1-3).

---

## ¿Qué pasa si el valor es 3?

Si `valor` es 3, entra en la condición del bloque `else` y se imprime:

X

---

## Salida esperada

Cada vez que ejecutes el programa, se imprimirá 1, 2 o X dependiendo del número aleatorio generado:

- Para `valor = 1`, salida: 1.
- Para `valor = 2`, salida: 2.
- Para `valor = 3`, salida: X.

**Ejercicio 2:** ¿Para qué sirve el siguiente programa? ¿Qué pasa si el valor es 3?

```
let valor = 0
valor = Math.randomRange(1, 3)

if (valor == 1) {
  basic.showLeds(`
    . . # . .
    . . # . .
    . . # . .
    . . # . .
    . . # . .
  `)
}
else if (valor == 2) {
  basic.showLeds(`
    # . . . #
    . # . # .
    . . # . .
    . # . # .
    # . . . #
  `)
}
else {
  basic.showLeds(`
    # # # # #
    . . . . #
    # # # # #
    # . . . .
    # # # # #
  `)
}
})
```

El programa **genera un número aleatorio entre 1 y 3** utilizando la función `Math.randomRange(1, 3)` y muestra un patrón LED diferente dependiendo del valor generado:

1. **Valor = 1:**
  - Muestra una línea vertical en el centro del panel LED.
2. **Valor = 2:**
  - Muestra un patrón de un corazón.
3. **Valor = 3:**
  - Muestra un patrón en forma de rectángulos y líneas en el panel LED.

---

### ¿Qué pasa si el valor es 3?

Si el valor aleatorio generado es **3**, el programa entra en el bloque `else` y ejecuta el siguiente patrón LED:

```
# # # # #
. . . . #
# # # # #
# . . . .
# # # # #
```

- Este patrón ilumina la matriz LED en forma de una figura con rectángulos y líneas.
-

## Funcionamiento del código

### Generación del número aleatorio:

```
valor = Math.randomRange(1, 3)
```

1.

- La función `Math.randomRange(1, 3)` genera un número aleatorio entero entre 1 y 3 (incluyendo ambos límites).

2. Estructura condicional:

- Dependiendo del valor generado, se ejecuta uno de los tres bloques de código:
  - Si `valor == 1`: Se muestra un patrón vertical.
  - Si `valor == 2`: Se muestra un corazón.
  - Si no es ninguno de los anteriores (en este caso, solo puede ser 3), se muestra el patrón final.

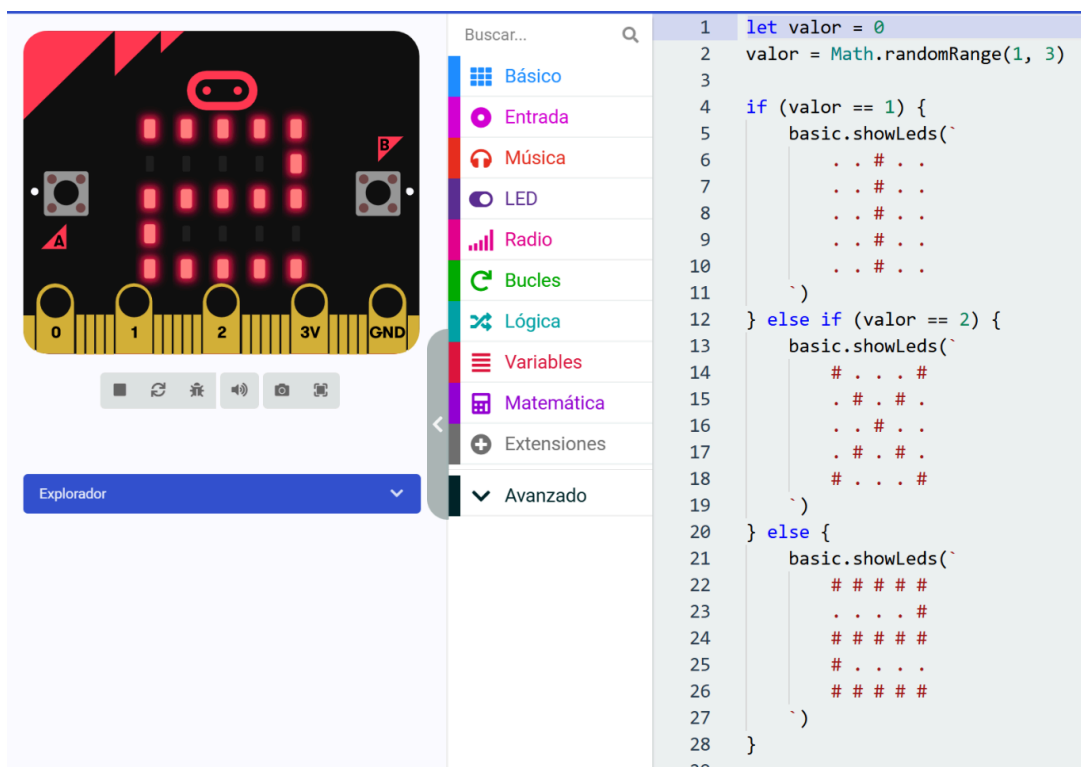
3. Uso de `basic.showLeds`:

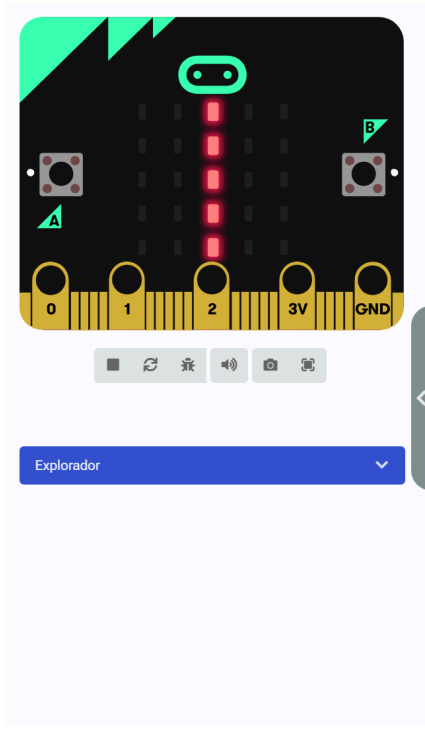
- La función `basic.showLeds` se utiliza para iluminar la matriz LED de 5x5 del Micro:bit con un diseño específico.

---

## Salida esperada

- Si `valor = 1`, se ilumina una línea vertical.
- Si `valor = 2`, se ilumina un corazón.
- Si `valor = 3`, se ilumina el patrón rectangular.

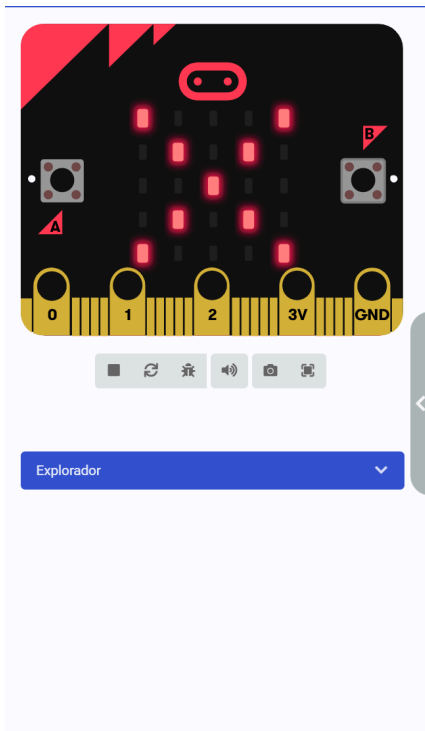




```

1  let valor = 0
2  valor = Math.randomRange(1, 3)
3
4  if (valor == 1) {
5      basic.showLeds(`
6          . . # . .
7          . . # . .
8          . . # . .
9          . . # . .
10         . . # . .
11     `)
12  } else if (valor == 2) {
13      basic.showLeds(`
14          # . . . #
15          . # . # .
16          . . # . .
17          . # . # .
18          # . . . #
19     `)
20  } else {
21      basic.showLeds(`
22          # # # # #
23          . . . . #
24          # # # # #
25          # . . . .
26          # # # # #
27     `)
28  }

```



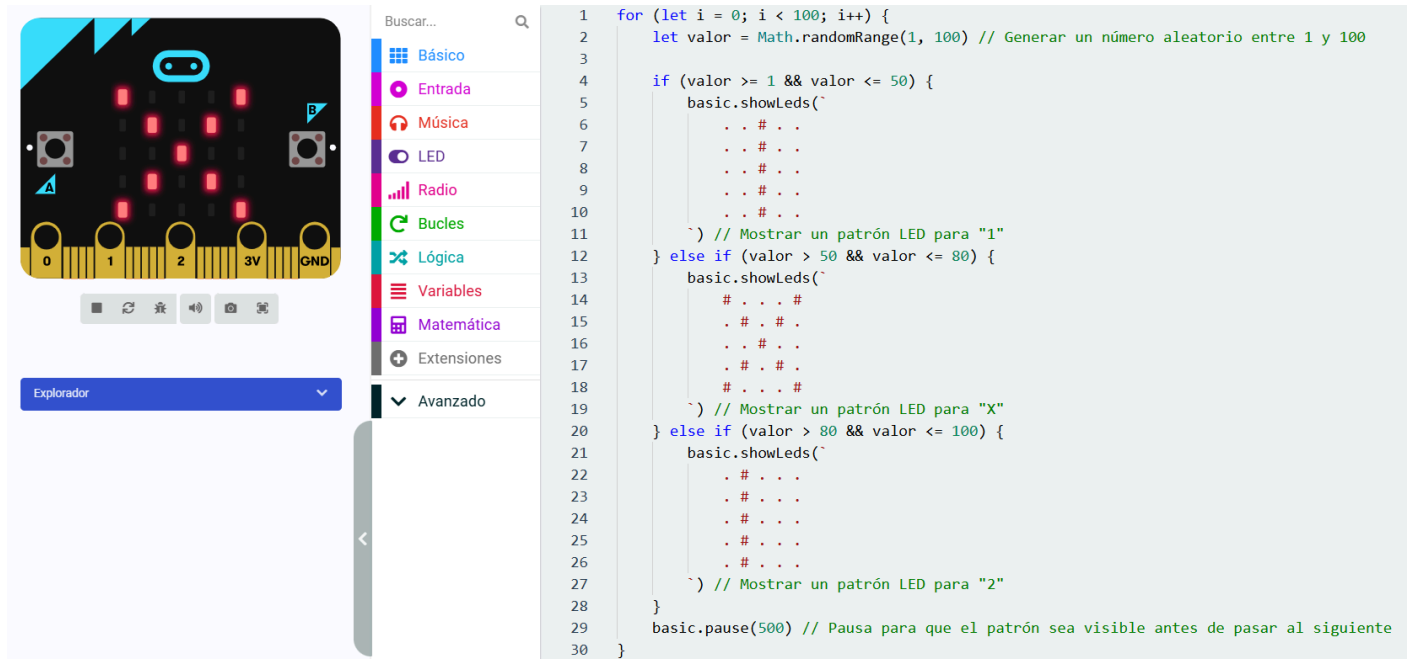
```

1  let valor = 0
2  valor = Math.randomRange(1, 3)
3
4  if (valor == 1) {
5      basic.showLeds(`
6          . . # . .
7          . . # . .
8          . . # . .
9          . . # . .
10         . . # . .
11     `)
12  } else if (valor == 2) {
13      basic.showLeds(`
14          # . . . #
15          . # . # .
16          . . # . .
17          . # . # .
18          # . . . #
19     `)
20  } else {
21      basic.showLeds(`
22          # # # # #
23          . . . . #
24          # # # # #
25          # . . . .
26          # # # # #
27     `)
28  }

```

### Ejercicio 3:

Modificar el anterior programa para que saque 100 numeros al azar, si el numero que sale es del 1 al 50 sale un **1**, si es del del 50 al 80 sale una **X** y de 80 a 100 sale un **2**



#### Ejercicio 4: ¿Qué realiza el siguiente código?

```

# Variables
numero1 = 2
numero2 = 3
resultado = 0

# Función para simular la acción al presionar un botón
def boton_click():
    global resultado
    resultado = numero1 + numero2
    print("El resultado es")
    print(resultado)

# Simulación de presionar boton (llamamos a la función aquí mismo)
boton_click()

```

Se definen tres variables:

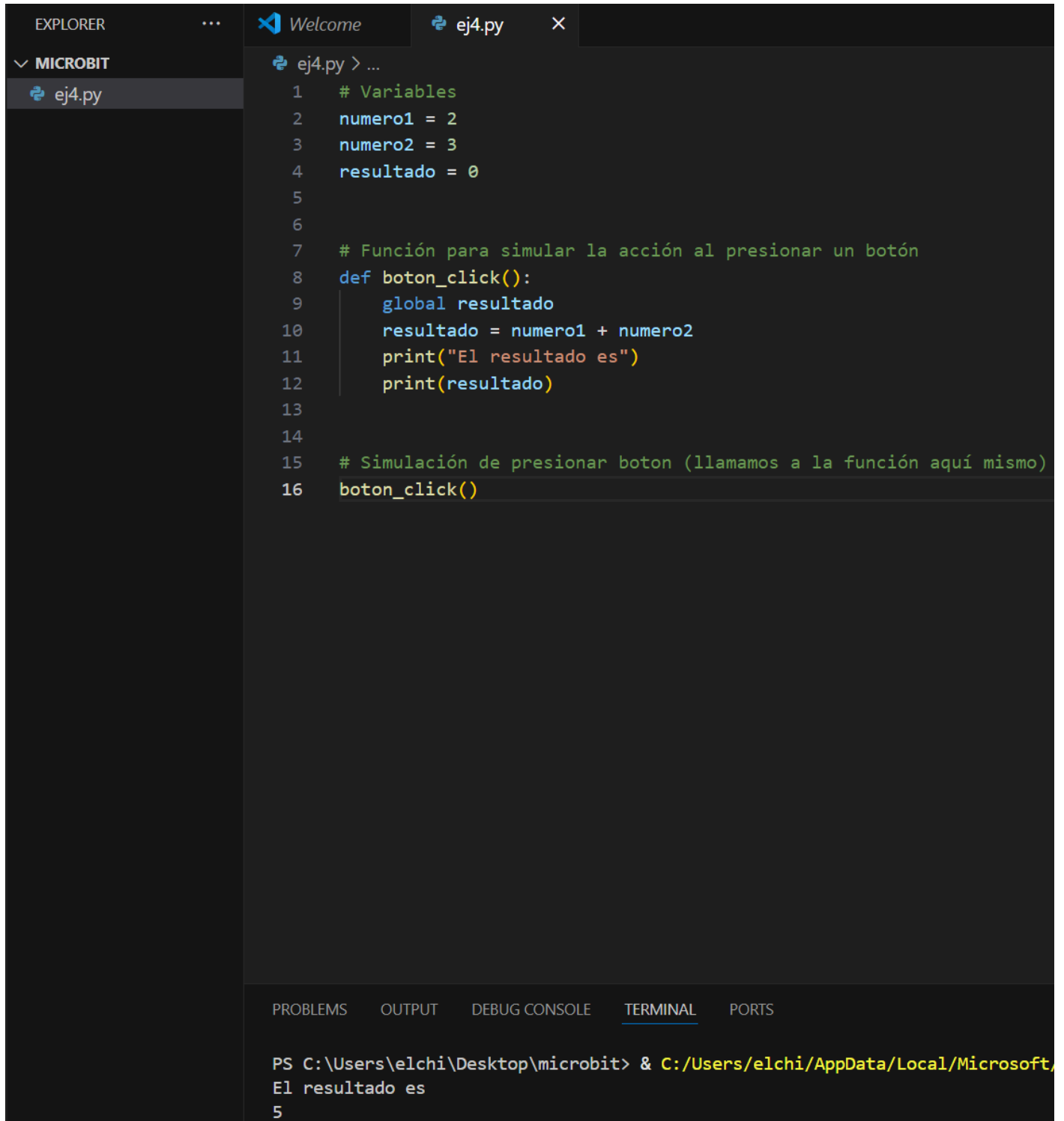
- `numero1` con el valor 2.
- `numero2` con el valor 3.
- `resultado` se inicializa en 0.

La función `boton_click()` simula lo que sucede al presionar un botón:

- Utiliza la palabra clave `global` para modificar la variable global `resultado`.
- Calcula la suma de `numero1` y `numero2`, y asigna el resultado a la variable `resultado`.
- Muestra en pantalla el texto "El resultado es" seguido del valor de `resultado`.

`boton_click()`

- Esta línea llama a la función `boton_click()`, simulando el evento de presionar un botón.



```
EXPLORER  ...  Welcome  ej4.py  X

MICROBIT
ej4.py

ej4.py > ...
1  # Variables
2  numero1 = 2
3  numero2 = 3
4  resultado = 0
5
6
7  # Función para simular la acción al presionar un botón
8  def boton_click():
9      global resultado
10     resultado = numero1 + numero2
11     print("El resultado es")
12     print(resultado)
13
14
15  # Simulación de presionar boton (llamamos a la función aquí mismo)
16  boton_click()

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\elchi\Desktop\microbit> & C:/Users/elchi/AppData/Local/Microsoft/
El resultado es
5
```

**Ejercicio 5:** a) ¿Qué realiza el siguiente código?

```
let numero1 = 2
let numero2 = 3
let resultado = 0

input.onPinPressed (TouchPin.P0,
    function () {
        resultado = numero1 + numero2

        basic.showString("El resultado es")
        basic.showNumber(resultado)
    }
)
```

Su propósito es calcular y mostrar la suma de dos números (`numero1` y `numero2`) cuando se toca el pin P0 del Micro:bit.

Se definen tres variables:

- `numero1` y `numero2` contienen los valores `2` y `3`, respectivamente.
- `resultado` se inicializa en `0`.

**Evento `onPinPressed`:** Se asocia al pin P0 del Micro:bit.

- Cuando el pin P0 es tocado, se dispara el evento y ejecuta la función asociada.

Dentro de la función:

1. Calcula la suma de `numero1` y `numero2`, y almacena el resultado en `resultado`.
2. Muestra el texto "El resultado es" en la pantalla LED del Micro:bit.
3. Muestra el número almacenado en `resultado`.

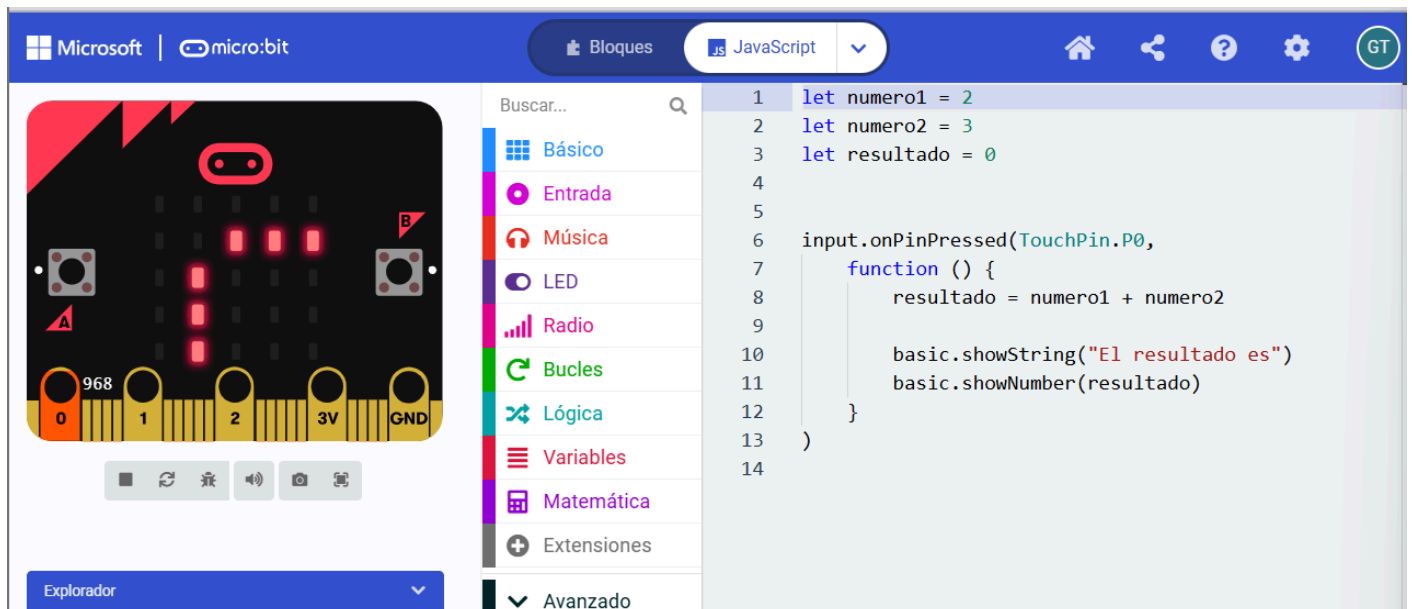
b) ¿Cuándo se dispara la función?

La función se dispara **cuando se toca o presiona el pin P0** en el Micro:bit. Este evento es controlado por:

```
input.onPinPressed(TouchPin.P0, function () { ... })
```

El pin P0 es uno de los pines táctiles del Micro:bit.

Cuando se detecta que el pin P0 ha sido activado por contacto físico o una conexión eléctrica, se ejecuta el bloque de código dentro del evento.



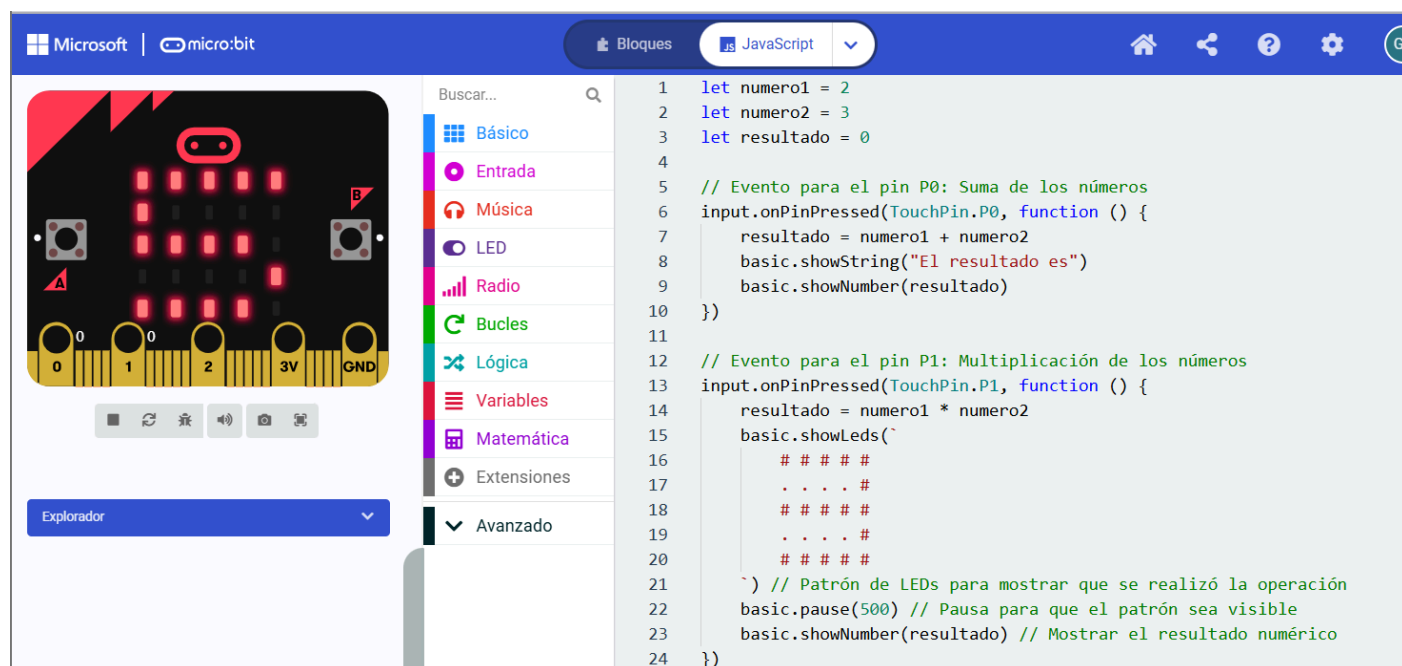
### Ejercicio 6:

Añadir código de una nueva función `input.onPinPressed2` al ejercicio anterior para que, al pulsar P1 realice la multiplicación de los valores que hay en `numero1` y `numero2`.

El resultado debe aparecer en pantalla con `showleds`

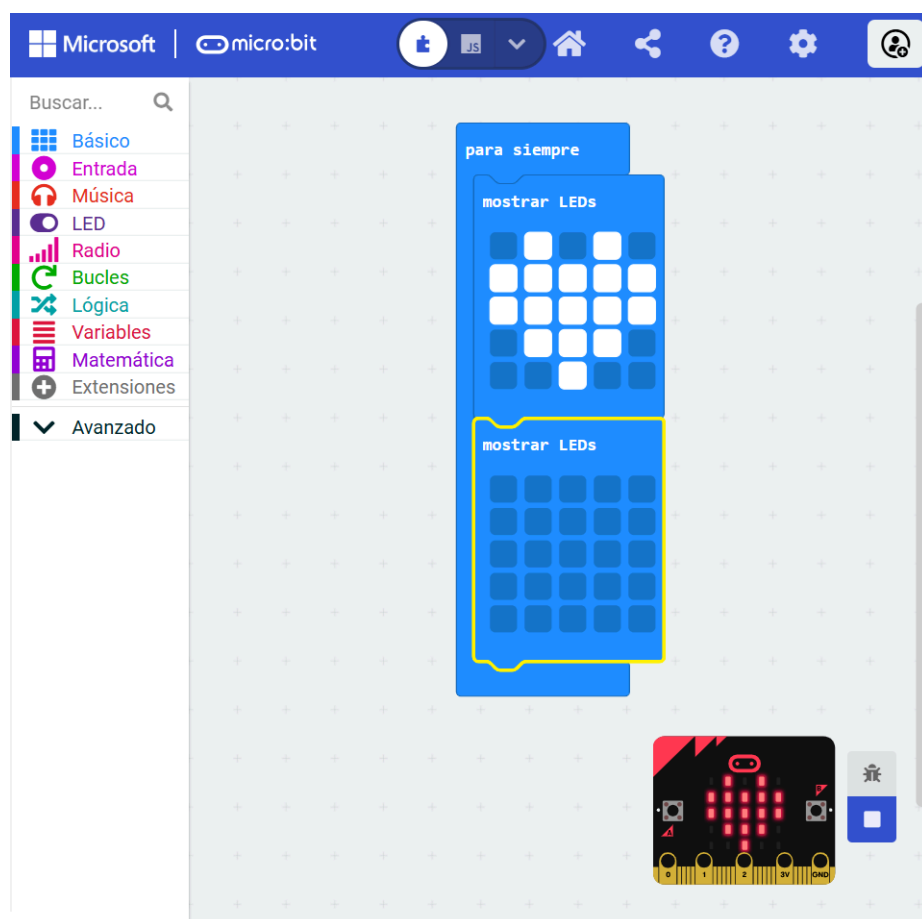


nueva función que se activa al pulsar el pin P1, realizando la multiplicación de los valores numero1 y numero2, y mostrando el resultado en la pantalla con un diseño de LEDs usando basic.showLeds.

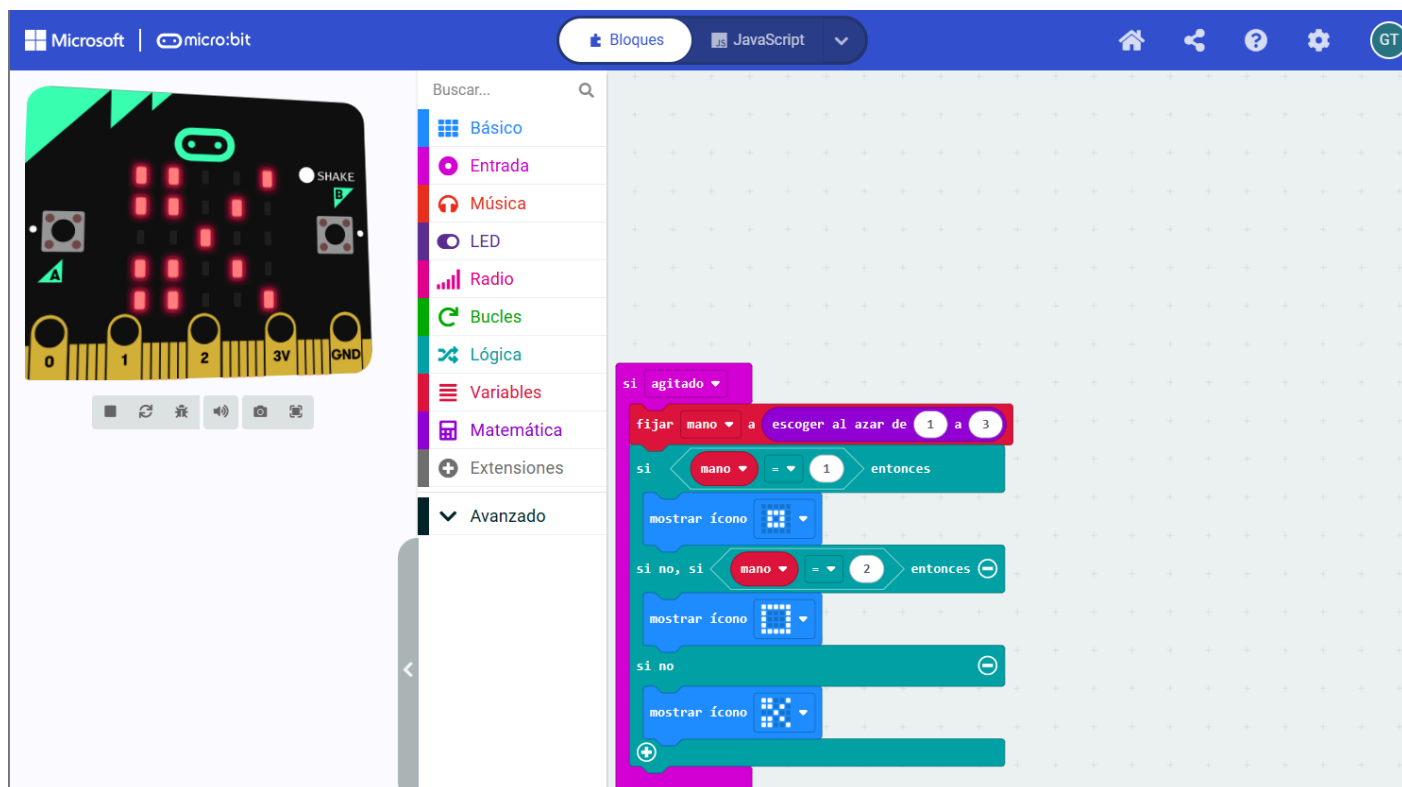


**Ejercicio 7: ONLINE MICROBIT** - <https://makecode.microbit.org/>

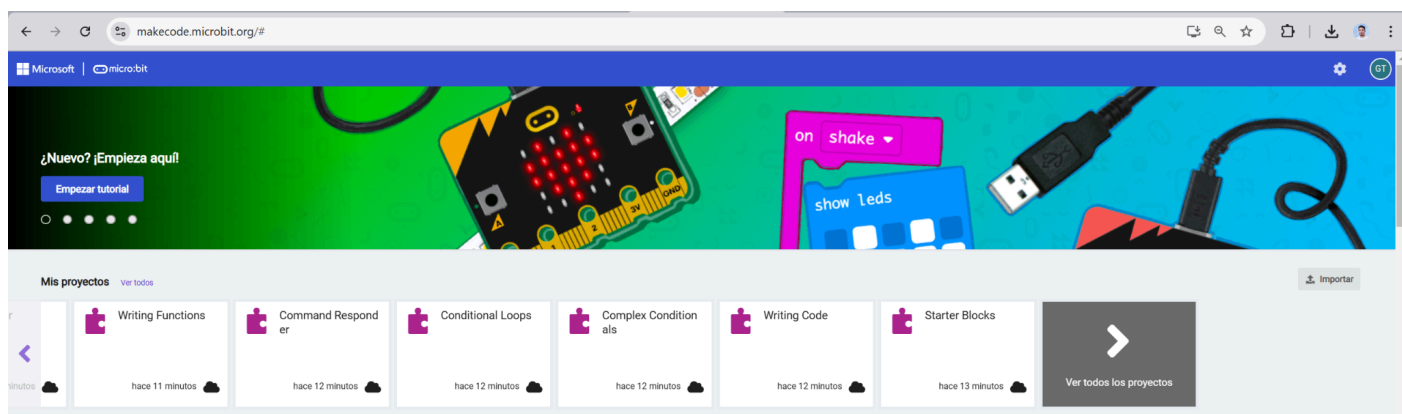
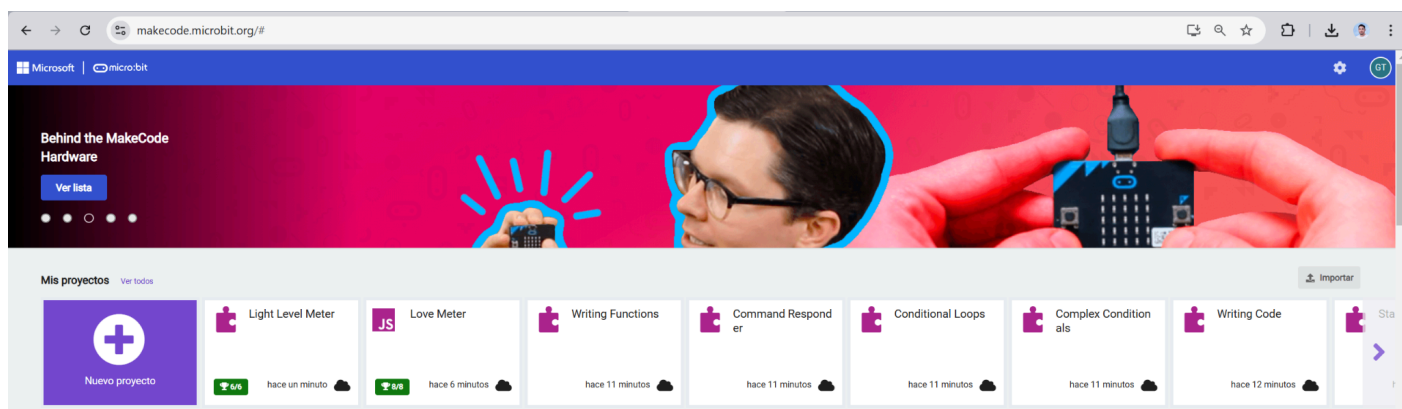
### 1) Realizar el Tutorial : FLASHING HEART



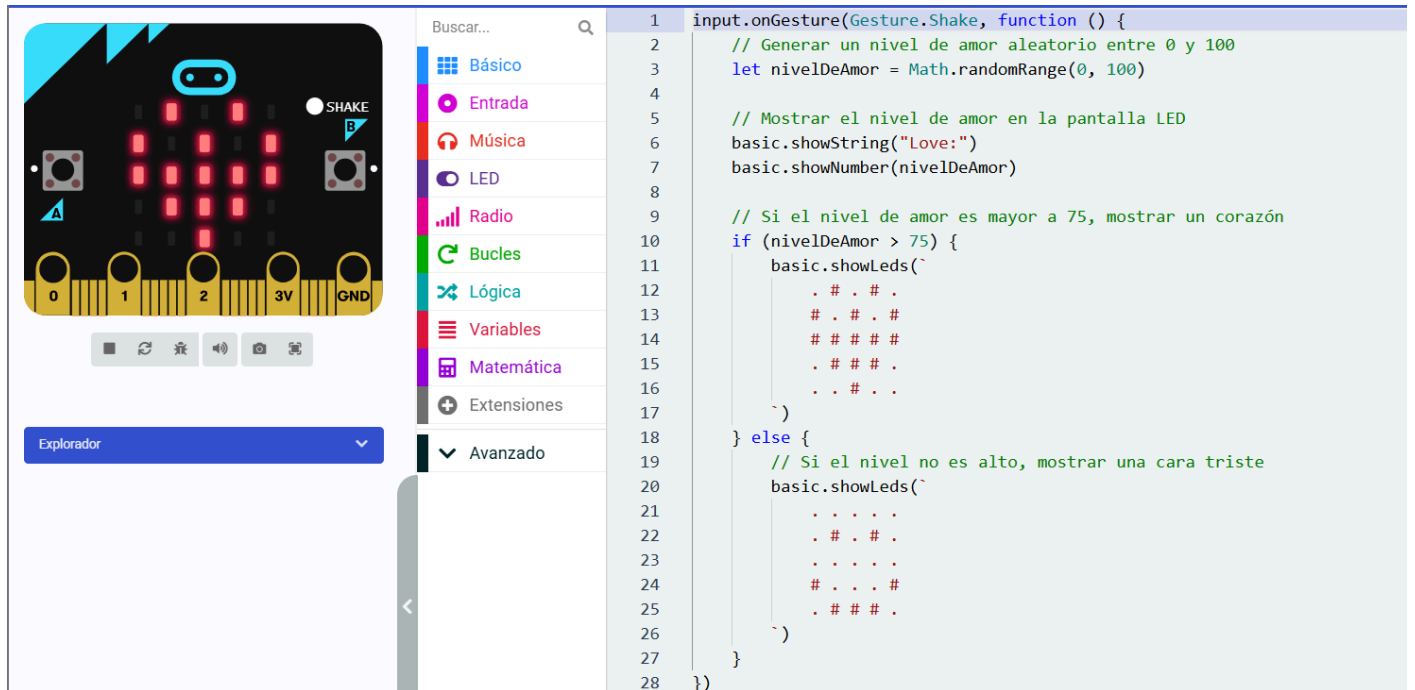
## 2) Realizar el Tutorial : En JUEGOS (GAMES) Rock Paper Scissors en BLOQUES



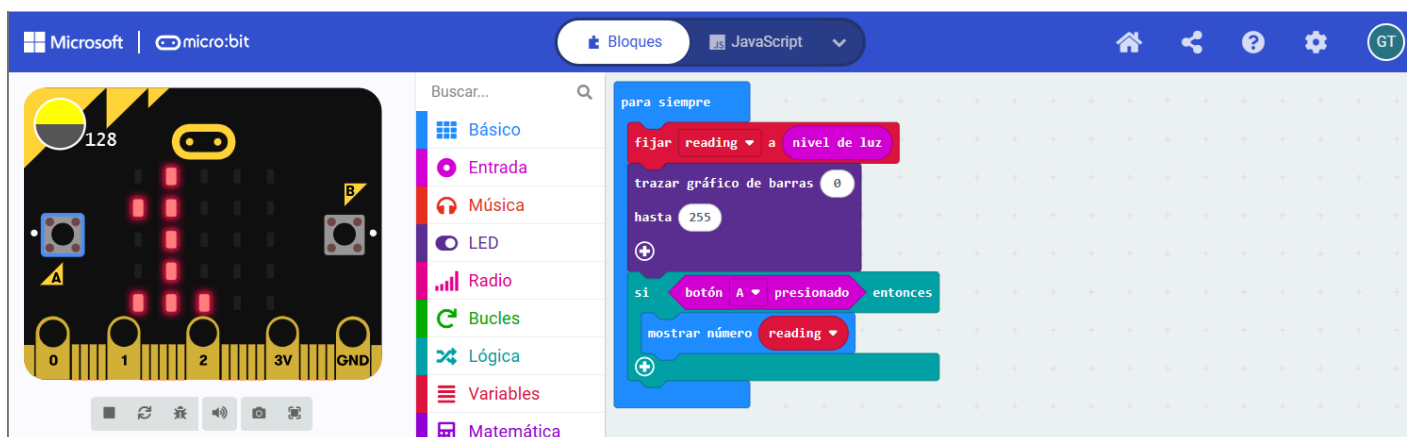
## 3) Realizar los tutoriales “ BLOCKS TO JAVASCRIPT “



## 4) Realizar el tutorial LoveMeter ❤️, en Javascript



5) Realizar el tutorial *Light Level Meter* <https://makecode.microbit.org/projects/light-level-meter>



## Ejercicio 8: Proyecto Programación Hardware

INVENTAR nuevo proyecto y **plantear el enunciado al profesor antes** de realizarlo en código para Micro:bit

### Propuesta de Proyecto de Programación Hardware para Micro:bit

Título del Proyecto: "Estación Meteorológica Básica con Micro:bit"

---

#### Enunciado del Proyecto

**Objetivo:** Crear un programa para el Micro:bit que actúe como una **estación meteorológica básica**, capaz de medir y mostrar:

1. **Temperatura ambiente** usando el sensor integrado del Micro:bit.
2. **Nivel de luz** utilizando el sensor de luz.
3. **Simulación de una alerta de humedad alta** (simulada con un botón o entrada táctil).

El proyecto incluirá interacciones con botones y una salida visual en la pantalla LED del Micro:bit.

---

#### Detalles del funcionamiento

1. **Medición de temperatura:**
  - Cada vez que se presione el botón **A**, el Micro:bit mostrará la temperatura ambiente en grados Celsius en la pantalla LED.
2. **Medición de luz:**
  - Al presionar el botón **B**, el Micro:bit mostrará el nivel de luz detectado en una escala del 0 al 100, donde 0 es oscuro y 100 es muy luminoso.
3. **Simulación de humedad alta:**
  - Si se toca el pin **P0**, se simulará una condición de alta humedad y el Micro:bit mostrará un ícono de alerta (un paraguas o gotas de agua).
4. **Alarma visual:**
  - Si la temperatura supera un umbral de 30°C, el Micro:bit mostrará un ícono de fuego (🔥) en la pantalla LED.

#### OPCIONAL: ¿Qué significa programar "Functional"?

Pensar en "módulos" que realizan un trabajo y devuelven un valor (sin afectar a ninguna otra parte del código)

La programación funcional es un paradigma de programación en el que el enfoque principal está en **usar funciones puras** para construir programas. Una **función pura**:

1. Siempre devuelve el mismo resultado si se le dan los mismos argumentos (sin efectos secundarios).
2. No modifica variables globales ni afecta el estado externo.

El objetivo es dividir el código en **módulos independientes** que realizan tareas específicas, devolviendo un valor sin afectar el resto del programa.

#### DA IGUAL EL LENGUAJE UTILIZADO.

PYTHON

```
def comprueba_ciudad(ciudad_bien, ciudad_usuario):
    if ciudad_bien.lower() == ciudad_usuario.lower():
        print("¡La respuesta es correcta!")
        return True
    else:
```

```

        print("Lo siento, la respuesta es incorrecta.")
        return False

def introduce_respuesta(dato_pais):
    pregunta = f"¿Cuál es la capital de {dato_pais}? "
    respuesta = input(pregunta)
    return respuesta

def main():
    puntos = 0

    respuesta_usuario = introduce_respuesta("España")
    if comprueba_ciudad("Madrid", respuesta_usuario):
        puntos += 1

    respuesta_usuario = introduce_respuesta("Francia")
    if comprueba_ciudad("París", respuesta_usuario):
        puntos += 1

    print(f"Puntos totales: {puntos}")

if __name__ == "__main__":
    main()

```

### ***Ejemplo en Java:***

```

import java.util.Scanner;

class Input {

    public static boolean compruebaCiudad(String ciudadBien, String ciudadUsuario){
        if (ciudadBien.equals(ciudadUsuario)) {
            System.out.println("¡La respuesta es correcta!");
            return true;
        } else {
            System.out.println("Lo siento, la respuesta es incorrecta.");
            return false;
        }
    }

    public static String introduceRespuesta(String datoPais) {
        Scanner introduceRespuesta = new Scanner(System.in);
        String pregunta = "¿Cuál es la capital de " + datoPais + " ?";
        System.out.print(pregunta);
        String respuesta = introduceRespuesta.nextLine();
        return respuesta;
    }

    public static void main(String[] args) {
        int puntos=0;
        String respuestaUsuario;
        boolean compruebaRespuesta;

        respuestaUsuario = introduceRespuesta("España");
        compruebaRespuesta = compruebaCiudad("Madrid", respuestaUsuario);
        if(compruebaRespuesta) { puntos++; }

        respuestaUsuario = introduceRespuesta("Francia");
        compruebaRespuesta = compruebaCiudad("Paris", respuestaUsuario);
        if(compruebaRespuesta) { puntos++; }

        System.out.print(puntos);
    }
}

```

}  
}