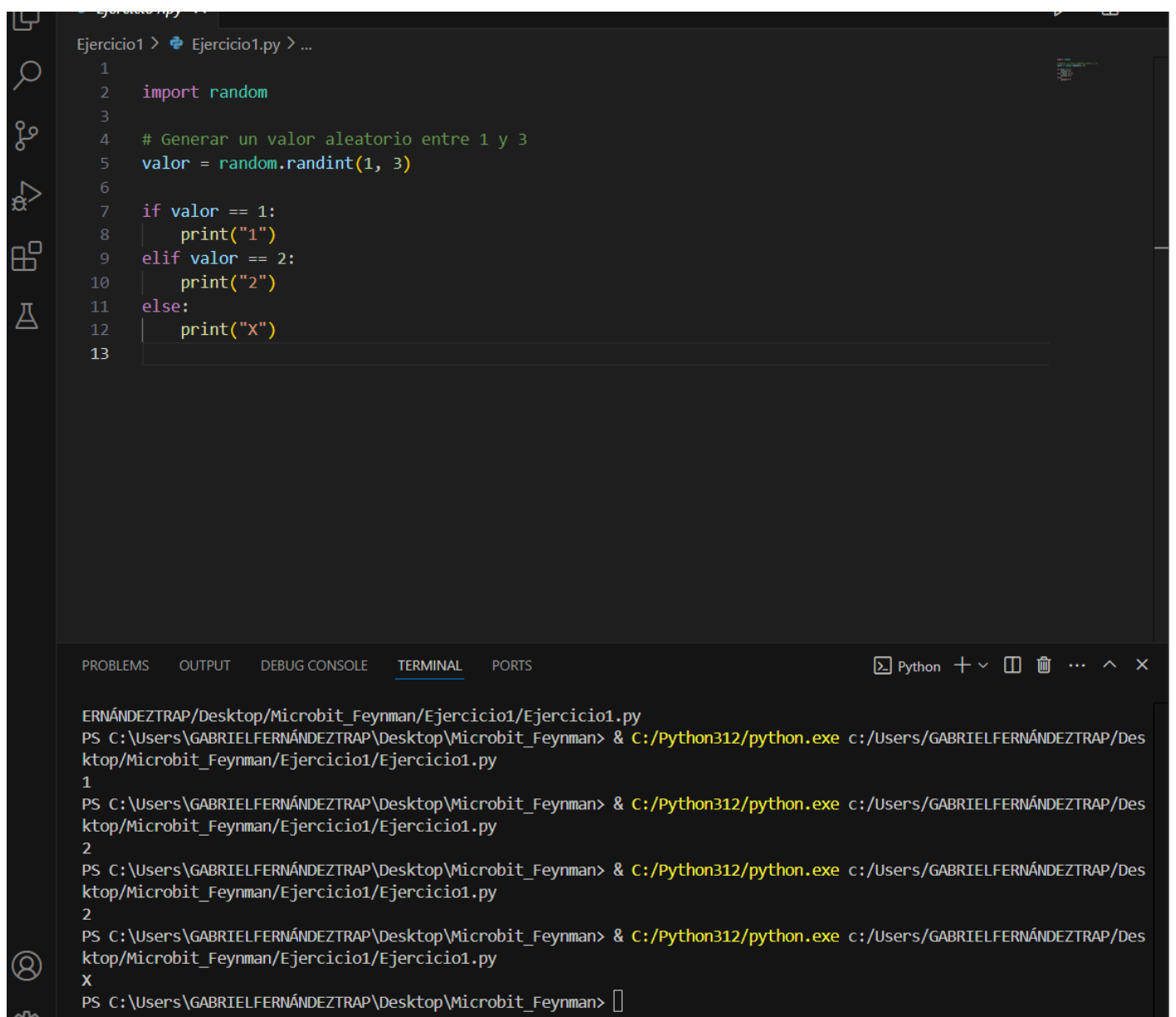


Ejercicios Feynman-Microbit

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 file named 'Ejercicio1.py'. The code in the editor is as follows:

```
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
```

The terminal at the bottom shows the execution of the program three times, each time using the command: `PS C:\Users\GABRIELFERNÁNDEZTRAP\Desktop\Microbit_Feynman> & C:/Python312/python.exe c:/Users/GABRIELFERNÁNDEZTRAP/Desktop/Microbit_Feynman/Ejercicio1/Ejercicio1.py`. The output of the program is shown on the line immediately following each command:

```
1
2
2
X
```

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(`
    # # # # #
    . . . . #
    # # # # #
    # . . . .
    # # # # #
  `)
}
})
```

Ejercicio 1: Contador de pasos básico

Un ejercicio inicial perfecto para **Micro:bit** y programación es crear un **contador de pasos** usando el acelerómetro integrado. Este ejercicio utiliza el entorno de bloques de MakeCode, es visual y fácil.

1. Introducir el entorno de programación con bloques (MakeCode).
2. Familiarizarse con conceptos básicos como:
 - Variables.
 - Condicionales.
 - Sensores integrados (acelerómetro).

Instrucciones paso a paso:

1. **Crear el proyecto en MakeCode:**
 - Abre MakeCode para Micro:bit.
 - Haz clic en "Nuevo proyecto" y nómbralo "Contador de pasos".
2. **Definir la variable:**
 - Ve a la categoría "Variables" y haz clic en "Crear una variable".
 - Llámala "pasos"
 - Arrastra el bloque establecer pasos a 0 dentro del bloque al iniciar.
3. **Detectar movimiento:**
 - Ve a la categoría "Entrada" y selecciona el bloque al agitar.
 - Dentro de este bloque, arrastra el bloque cambiar pasos por 1 desde la categoría "Variables".
4. **Mostrar los pasos en la pantalla:**

- Desde la categoría "Básico", arrastra el bloque mostrar número y ponlo en el evento al presionar botón A.
- Configúralo para mostrar la variable pasos.

5. **Opcional: Resetear el contador (botón B):**

- Agrega otro bloque al presionar botón B desde la categoría "**Entrada**".
- Dentro, arrastra el bloque establecer pasos a 0 desde "**Variables**".

Resumen Código final (en bloques):

- **Al iniciar:** Establecer pasos en 0.
- **Al agitar:** Incrementar pasos en 1.
- **Al presionar botón A:** Mostrar el valor de pasos en la pantalla LED.
- **Al presionar botón B:** Restablecer pasos a 0.

Explicación del ejercicio:

- El Micro:bit tiene un **acelerómetro** que detecta el movimiento. Este sensor se utiliza para aumentar el contador cada vez que el dispositivo se agita.
- Al presionar los botones A o B, los alumnos interactúan directamente con el dispositivo para consultar o reiniciar el contador.

Ejercicio 1b: Contador de pasos básico con Python

Un ejercicio similar al contador de pasos, pero en **Python** (usando Micro:bit y su editor), puede ser ideal para introducir conceptos básicos de programación como variables, condicionales y entrada/salida.

Instrucciones paso a paso:

1. Preparar el entorno:

- Abre el editor Python para Micro:bit (MicroPython en MakeCode).
- Crea un nuevo proyecto y guárdalo como contador_pasos.py.

código básico:

```
from microbit import *

# Inicializar el contador de pasos
pasos = 0

while True:
    # Detectar si el Micro:bit está siendo agitado
    if accelerometer.was_gesture("shake"):
        pasos += 1 # Incrementar el contador

    # Mostrar pasos al presionar botón A
    if button_a.is_pressed():
        display.scroll(str(pasos))

    # Reiniciar el contador al presionar botón B
    if button_b.is_pressed():
        pasos = 0
        display.show("R") # Mostrar una "R" como confirmación
```

Explicación del código:

1. Importar la librería:

- `from microbit import *` incluye todas las funciones necesarias para interactuar con el Micro:bit.

2. Inicializar la variable:

- `pasos = 0` define y establece el contador en 0.

3. Bucle principal:

- `while True:` asegura que el programa se ejecute continuamente.

4. Detectar movimiento:

- `accelerometer.was_gesture("shake")` detecta si el Micro:bit ha sido agitado.
- Si se detecta el movimiento, se incrementa el contador.

5. Mostrar los pasos:

- `button_a.is_pressed()` muestra el número de pasos en la pantalla LED al presionar el botón A.

6. Reiniciar el contador:

- `button_b.is_pressed()` reinicia el contador y muestra una confirmación (R) en la pantalla.

Ejercicio 2: : Generador de resultados para una quiniela

Objetivo:

1. Familiarizar a los alumnos con:
 - Uso de botones (A y B).
 - Mostrar caracteres (1, X, 2) en la pantalla LED.
 - Generación de resultados aleatorios.
2. Introducir el concepto de **aleatoriedad** en programación

Enunciado:

Programa el Micro:bit para que sea un **generador de quinielas**. Al presionar el botón **A**, mostrará un resultado aleatorio entre 1, X y 2. Si presionas el botón **B**, limpiará la pantalla.

Instrucciones:

1. Usa la función de números aleatorios para seleccionar entre tres opciones: 1, X, 2.
2. Muestra el resultado en la pantalla LED al presionar el botón A.
3. Al presionar el botón B, limpia la pantalla para preparar el próximo resultado.

Ejemplo de código en bloques (MakeCode):

1. Ve a la categoría **Entrada** y selecciona al presionar botón A.
2. Usa la categoría **Matemáticas** para agregar elegir aleatoriamente.
 - Configúralo para elegir entre los valores 1, "X" y 2.
3. Usa la categoría **Básico** para mostrar el número o carácter en pantalla.
4. Agrega un bloque al presionar botón B y dentro pon borrar pantalla.

El programa debería verse así:

- **Cuando se presiona A:** Mostrar un valor aleatorio 1, X o 2.
- **Cuando se presiona B:** Borrar la pantalla.

```
from microbit import *
import random

while True:
    # Mostrar un resultado aleatorio al presionar A
    if button_a.is_pressed():
        resultado = random.choice(["1", "X", "2"])
        display.show(resultado)

    # Limpiar la pantalla al presionar B
    if button_b.is_pressed():
        display.clear()
```

Ejercicio 3: Máquina del amor - Medidor de compatibilidad amorosa

Objetivo:

1. Usar el sensor de temperatura del Micro:bit.
2. Introducir conceptos básicos:
 - Lectura de sensores.
 - Generación de puntuaciones a partir de datos.
 - Mostrar resultados en pantalla.

Enunciado:

Construye una máquina del amor con tu Micro:bit. La máquina medirá tu compatibilidad amorosa basándose en la temperatura de tu mano o en la estabilidad del dispositivo. Al presionar el botón **A**, mostrará un número entre 0 y 100 como tu "puntuación de amor". Usa la función de temperatura como entrada para hacer el resultado más interesante.

Instrucciones para los alumnos:

1. Usa el sensor de temperatura del Micro:bit.
2. Calcula una puntuación "amorosa" basándote en la temperatura medida.
3. Muestra el resultado al presionar el botón **A**.
4. Opcionalmente, agrega un mensaje romántico si la puntuación es alta (por ejemplo, mayor a 75).

Ejemplo de código en bloques (MakeCode):

1. Usa la categoría **Entrada** para leer la temperatura (temperatura ambiente).
2. Genera una puntuación "amorosa" basada en la temperatura:
 - Multiplica la temperatura por un valor, por ejemplo, 4, y ajusta con una fórmula para que el rango esté entre 0 y 100.
3. Al presionar el botón A:
 - Muestra la puntuación en la pantalla.
 - Si la puntuación es alta, muestra un icono de corazón.
4. Opcionalmente, al presionar el botón B, limpia la pantalla.

Código en Python:

```
from microbit import *

while True:
    # Al presionar el botón A, calcular puntuación
    if button_a.is_pressed():
        temperatura = temperature() # Leer temperatura ambiente
        puntuacion = (temperatura * 4) % 101 # Generar puntuación entre 0 y 100

        # Mostrar puntuación en pantalla
        display.scroll(str(puntuacion))

        # Mostrar un corazón si la puntuación es alta
        if puntuacion > 75:
            display.show(Image.HEART)
        else:
            display.clear()

    # Limpiar pantalla con el botón B
    if button_b.is_pressed():
        display.clear()
```

Explicación del código:

1. **Lectura de la temperatura:**

temperature() lee la temperatura en grados Celsius del sensor del Micro:bit.

2. Cálculo de puntuación:

- $(\text{temperatura} * 4) \% 101$ convierte el rango de temperatura en una puntuación "amorosa" entre 0 y 100

3. Mostrar resultados:

- Usa display.scroll para mostrar la puntuación.
- Si la puntuación supera 75, muestra un corazón (Image.HEART).

4. Limpieza opcional:

- Limpia la pantalla con el botón **B** para iniciar una nueva medición.

Extensiones para hacerlo más interesante:

1. Añadir mensajes románticos

- Mostrar "HOT" si la puntuación es mayor a 90 o "MEH" si está por debajo de 30.

2. Sensores adicionales:

- Usar el acelerómetro para añadir "estabilidad" como otro factor en la puntuación.

3. Efectos visuales:

- Añadir una animación mientras se calcula la puntuación.

<https://chatgpt.com/share/674e476a-85b4-8009-a402-ccf808f7fe1a>