Programación Avanzada con Microbit

1. Objetivo

Los estudiantes aprenderán conceptos avanzados de programación utilizando la placa Microbit. Desarrollarán habilidades para trabajar con sensores como el acelerómetro, el sensor de temperatura y la brújula, además de manipular la intensidad luminosa, combinando estos conocimientos con bucles, condicionales y eventos utilizando tanto MICROPYTHON como bloques en MAKECODE.

2. Duración

2 horas

3. Materiales Necesarios

- Placas Microbit y cables USB para la conexión.
- Computadoras con acceso a Internet.
- Acceso a las plataformas de programación MICROYPYTHON y MAKECODE.
- Proyector o pantalla para mostrar ejemplos de programación.

4. Estructura de la Actividad

4.1. Desarrollo de la Actividad

4.1.1. Ejercicio 1: Acelerómetro

- **Planteamiento del Problema:** Crear un programa en la micro

que lea la inclinación en el eje X utilizando el acelerómetro. Si la inclinación es mayor a 200, se mostrará ">" en la pantalla LED; si es menor a -200, se mostrará "<"; y si está entre -200 y 200, se mostrará "-". Este proceso debe repetirse continuamente.

4.1.2. Ejercicio 2: Sensor de Temperatura

- **Planteamiento del Problema:** Crear un programa en la micro

que lea la temperatura ambiente utilizando el sensor integrado y la muestre en la pantalla LED cada segundo. El valor de la temperatura se actualiza continuamente, acompañado de la letra "C" para indicar que la medida está en grados Celsius.

4.1.3. Ejercicio 3: Intensidad Luminosa

- **Planteamiento del Problema:** Crear un programa sencillo utilizando la micro que permita medir la intensidad luminosa de una habitación y representarla como un diagrama de barras en la pantalla LED de la micro, de forma continua.

4.1.4. Ejercicio 4: Brújula

- **Planteamiento del Problema:** Crear un programa sencillo utilizando la micro que use su brújula integrada para mostrar en la pantalla LED la dirección cardinal hacia la que está orientada (Norte, Este, Sur u Oeste). El programa debe calibrar la brújula al inicio y luego mostrar continuamente la dirección en la pantalla.

4.1.5. Ejercicio 5: Acelerómetro y Brújula

- **Planteamiento del Problema:** Crear un programa en la micro que combine el uso de la brújula y el acelerómetro. El programa debe mostrar la dirección cardinal hacia la que está orientada la micro (Norte, Este, Sur, Oeste) y, al mismo tiempo, indicar si la micro está inclinada hacia adelante, hacia atrás o si está en posición neutral.

5. Rúbrica de Evaluación

Criterio	No Entrega	Mejorable	Bien	Excelente
Precisión y Exhaustividad	No se entrega trabajo. (0 puntos)	Faltan varios conceptos clave o las soluciones no cumplen con los objetivos. (5 puntos)	La mayoría de los conceptos están incluidos y las soluciones son funcionales con algunos errores menores. (8 puntos)	Todos los conceptos clave están incluidos y las soluciones son precisas, completas y funcionan correctamente. (10 puntos)

6. Objetivos de Desarrollo Sostenible (ODS) relacionados

- **ODS 4: Educación de Calidad:** Fomentar una comprensión profunda de la programación a través de la práctica y la resolución de problemas.
- **ODS 9: Industria, Innovación e Infraestructura:** Promover la innovación y la comprensión tecnológica mediante la programación de dispositivos físicos.

7. Principios del Diseño Universal para el Aprendizaje (DUA)

- **Múltiples medios de representación:** Uso de diferentes lenguajes de programación (MICROPYTHON y MAKECODE) para representar soluciones.
- **Múltiples medios de acción y expresión:** Permitir a los estudiantes elegir entre código o bloques para expresar su solución.
- **Múltiples medios de compromiso:** Involucrar a los estudiantes en la creación de programas basados en sus propias interpretaciones y creatividad.

8. Resultados Esperados

- Los estudiantes habrán creado varios programas que demuestran su comprensión de los conceptos avanzados de programación.
- Habrán demostrado habilidades en la escritura de código, la resolución de problemas y el pensamiento lógico.
- Habrán participado activamente en la discusión, mostrando su comprensión del tema y su capacidad para conectar ideas de programación con la práctica en hardware.

9. Soluciones

9.1. Solución para Ejercicio 1: Acelerómetro

9.1.1. MICROPYTHON

python Copiar código

```python
def on_forever():
    # Bucle que muestra la inclinación en el eje X
    x = input.acceleration(Dimension.X)
    if x > 200:
        basic.show_string(">")
    elif x < -200:
        basic.show_string("<")
    else:
        basic.show_string("-")

basic.forever(on_forever)
```

9.1.2. BLOQUES MAKECODE

plaintext Copiar código

```
[forever]
    ├─ [set x to input acceleration (x)]
    ├─ [if x > 200 then]
    │       └─ [show string ">"]
    ├─ [else if x < -200 then]
    │       └─ [show string "<"]
    └─ [else]
            └─ [show string "-"]
```

1. Usar un bloque "para siempre".
2. Dentro del bloque, agregar un "si... entonces... sino si... sino".
3. Primera condición: inclinación en X > 200 (mostrar ">").
4. Segunda condición: inclinación en X < -200 (mostrar "<").
5. Tercera condición: inclinación entre -200 y 200 (mostrar "-").
6. Pausar 100 ms.

9.2. Solución para Ejercicio 2: Sensor de Temperatura

9.2.1. MICROPYTHON

python Copiar código

```python
def on_forever():
    # Bucle que muestra la temperatura cada segundo
    temp = input.temperature()
    basic.show_string(str(temp) + "C")
    basic.pause(1000)

basic.forever(on_forever)
```

9.2.2. BLOQUES MAKECODE

plaintext Copiar código

```
[por siempre]
    ├─ [establecer temperatura a temperatura (°C)]
    ├─ [mostrar cadena (temperatura + "C")]
    └─ [pausar (ms) 1000]
```

1. Usar un bloque "para siempre".
2. Dentro del bloque, "mostrar cadena" con el valor de la temperatura concatenado con "C".
3. Pausar 1000 ms.

9.3. Solución para Ejercicio 3: Intensidad Luminosa

9.3.1. MICROPYTHON

python Copiar código

```python
def on_forever():
    # Obtener el nivel de luz
    luz = input.light_level()

    # Mostrar el nivel de luz en la pantalla LED
    basic.show_number(luz)

# Iniciar la ejecución continua de la función on_forever
basic.forever(on_forever)
```

9.3.2. BLOQUES MAKECODE

plaintext Copiar código

```
[para siempre]
    ├─ [fijar luz a nivel de luz]
    └─ [trazar gráfico de barras luz hasta 255]
```

1. Usar un bloque "para siempre".
2. Dentro del bloque, medir la intensidad de luz y almacenarla.
3. Utilizar un bloque "mostrar barra" con el valor de la luz.
4. Pausar 100 ms.

9.4. Solución para Ejercicio 4: Brújula

9.4.1. MICROPYTHON

python Copiar código

```python
# Calibrar la brújula al inicio
input.calibrate_compass()

def on_forever():
    # Obtener el ángulo de la brújula
    angulo = input.compass_heading()

    # Mostrar la dirección cardinal según el ángulo
    if angulo < 45 or angulo > 315:
        basic.show_string("N")
    elif angulo < 135:
        basic.show_string("E")
    elif angulo < 225:
        basic.show_string("S")
    else:
        basic.show_string("O")

# Iniciar la ejecución continua de la función on_forever
basic.forever(on_forever)
```

9.4.2. BLOQUES MAKECODE

plaintext Copiar código

```
[al iniciar]
    └─ [calibrar brújula]

[para siempre]
    └─ [fijar angulo a dirección de la brújula (°)]
    └─ [si angulo < 45 o angulo > 315 entonces]
            └─ [mostrar cadena 'N']
        [si no, si angulo < 135 entonces]
            └─ [mostrar cadena 'E']
        [si no, si angulo < 225 entonces]
            └─ [mostrar cadena 'S']
        [si no]
            └─ [mostrar cadena 'O']
```

1. Utilizar un bloque "calibrar brújula" al inicio.
2. Usar un bloque "para siempre".
3. Dentro del bloque, usar "si... entonces... sino si... sino" para mostrar "N", "E", "S", o "W" basado en el valor de la dirección.
4. Pausar 100 ms.

9.5. Solución para Ejercicio 5: Acelerómetro y Brújula

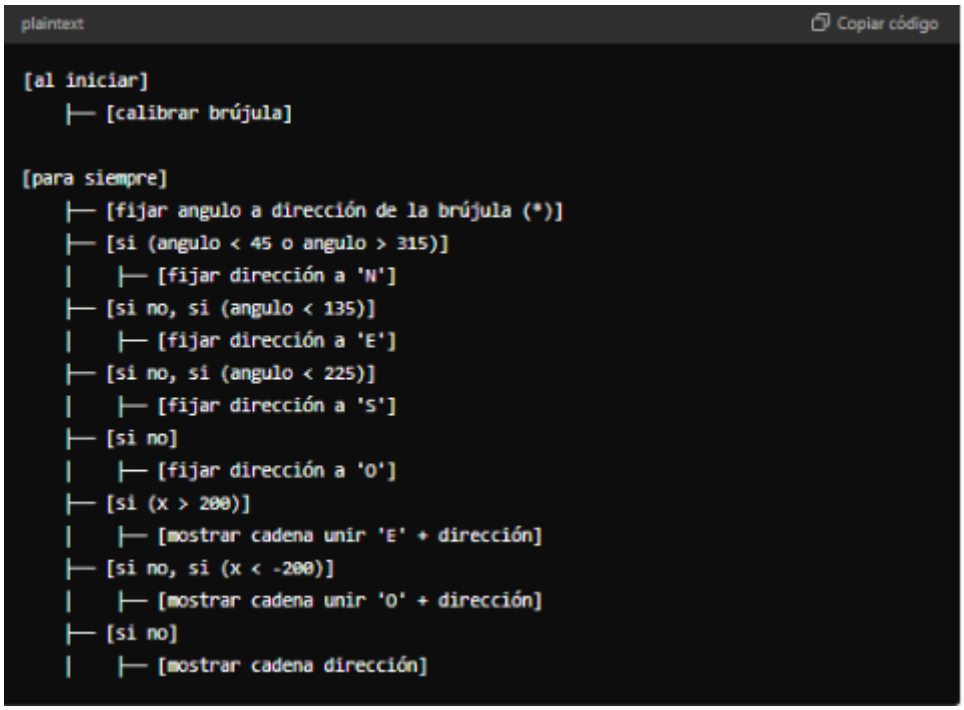
9.5.1. MICROPYTHON

python Copiar código

```python
# Calibrar la brújula al inicio
input.calibrate_compass()

def on_forever():
    # Obtener la inclinación en el eje X y la dirección de la brújula
    x = input.acceleration(Dimension.X)
    angulo = input.compass_heading()

    # Determinar la dirección cardinal
    if angulo < 45 or angulo > 315:
        direccion = "N"
    elif angulo < 135:
        direccion = "E"
    elif angulo < 225:
        direccion = "S"
    else:
        direccion = "O"

    # Determinar la inclinación y mostrar la dirección con el símbolo correspondiente
    if x > 200:
        basic.show_string(direccion + " >")
    elif x < -200:
        basic.show_string(direccion + " <")
    else:
        basic.show_string(direccion + " -")

# Iniciar la ejecución continua de la función on_forever
basic.forever(on_forever)
```

9.5.2. BLOQUES MAKECODE

1. Utilizar un bloque "calibrar brújula" al inicio.
2. Usar un bloque "para siempre".
3. Dentro del bloque, medir la dirección y la inclinación.
4. Usar "si... entonces... sino si... sino" para mostrar la dirección cardinal y la inclinación ("F", "B" o "-").
5. Pausar 100 ms.



ANEXOS

1. Guía de conceptos clave para la programación en Microbit.
2. Ejemplos de códigos y soluciones utilizando MICROPYTHON y MAKECODE.