Programación con el Robot Maqueen

1. Objetivo

Los estudiantes aprenderán a programar el robot Maqueen utilizando la micro. Desarrollarán habilidades para controlar el robot mediante el uso de sensores y motores, aplicando conceptos de programación como bucles, condicionales y eventos. Utilizarán tanto MICROPYTHON como bloques en MAKECODE para resolver diferentes retos de programación.

2. Duración

2 horas

3. Materiales Necesarios

- Robots Maqueen y micro
- Placas micro 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: Primeros pasos con Maqueen

- **Planteamiento del Problema:** El robot Maqueen debe avanzar hacia adelante a máxima velocidad durante 1 segundo, detenerse por medio segundo, luego retroceder a máxima velocidad durante 1 segundo, y finalmente detenerse nuevamente. Esta secuencia de movimientos puede utilizarse para realizar maniobras simples, como simular un movimiento de ida y vuelta.

4.1.2. Ejercicio 2: Luces y música con Maqueen

- **Planteamiento del Problema:** Controlar el robot Maqueen para que reproduzca un sonido, encienda ambos LEDs, espere un momento, apague los LEDs, avance a una velocidad moderada, y luego se detenga antes de repetir el ciclo.

4.1.3. Ejercicio 3: Trompos con Maqueen

- **Planteamiento del Problema:** El robot Maqueen debe realizar un ciclo de movimiento donde el motor izquierdo avance mientras el motor derecho retrocede, luego detenerse, y después realizar un movimiento inverso en el que el motor izquierdo retroceda y el motor derecho avance. Este ciclo se repetirá de manera continua.

4.1.4. Ejercicio 4: Evita obstáculos

- **Planteamiento del Problema:** El robot Maqueen debe avanzar hacia adelante, detectar obstáculos usando su sensor ultrasónico, y evitar estos obstáculos. Si se detecta un obstáculo a menos de 10 cm, el robot debe retroceder y girar antes de continuar avanzando.

4.1.5. Ejercicio 5: Sigue líneas

- **Planteamiento del Problema:** El robot Maqueen debe seguir una línea negra utilizando los sensores de siguelíneas (patrol). Si ambos sensores detectan la línea, el robot avanza; si solo uno de los sensores detecta la línea, el robot gira hacia la dirección correspondiente para corregir su trayectoria.

4.1.6. Ejercicio 6: Pilla - Pilla

- **Planteamiento del Problema:** El robot Maqueen debe perseguir un objeto si lo detecta a menos de 20 cm de distancia. Si no detecta ningún objeto, el robot gira en una dirección aleatoria.

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 y control de robots.
- Habrán demostrado habilidades en la escritura de código, la resolución de problemas y el pensamiento lógico aplicados a la robótica.
- 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: Primeros pasos con Maqueen

9.1.1. MICROPYTHON

python Copiar código

```python
# Mover hacia adelante a velocidad máxima
maqueen.motor_run(maqueen.Motors.ALL, maqueen.Dir.CW, 255)

# Espera 1 segundo
basic.pause(1000)

# Detener los motores
maqueen.motor_stop(maqueen.Motors.ALL)

# Espera medio segundo
basic.pause(500)

# Mover hacia atrás a velocidad máxima
maqueen.motor_run(maqueen.Motors.ALL, maqueen.Dir.CCW, 255)

# Espera 1 segundo
basic.pause(1000)

# Detener los motores
maqueen.motor_stop(maqueen.Motors.ALL)
```

9.1.2. BLOQUES MAKECODE

plaintext Copiar código

```
[para siempre]
    ├─ [Motor: ambos, sentido avanzar, velocidad 255]
    ├─ [pausa (ms): 1000]
    ├─ [Parar motor: ambos]
    ├─ [pausa (ms): 500]
    ├─ [Motor: ambos, sentido retroceder, velocidad 255]
    ├─ [pausa (ms): 1000]
    └─ [Parar motor: ambos]
```

1. Usar un bloque "al iniciar".
2. Configurar los motores para avanzar a máxima velocidad durante 1 segundo.
3. Detener los motores durante 0.5 segundos.
4. Configurar los motores para retroceder a máxima velocidad durante 1 segundo.
5. Detener los motores.

9.2. Solución para Ejercicio 2: Luces y música con Maqueen

9.2.1. MICROPYTHON

```python
def on_forever():
    music.play(music.string_playable("G B A G C5 B A B ", 120), music.PlaybackMode.UNTIL_DO
    maqueen.write_led(maqueen.LED.LED_LEFT, maqueen.LEDswitch.TURN_ON)
    maqueen.write_led(maqueen.LED.LED_RIGHT, maqueen.LEDswitch.TURN_ON)
    basic.pause(2000)
    maqueen.write_led(maqueen.LED.LED_LEFT, maqueen.LEDswitch.TURN_OFF)
    maqueen.motor_run(maqueen.Motors.ALL, maqueen.Dir.CW, 50)
    maqueen.write_led(maqueen.LED.LED_RIGHT, maqueen.LEDswitch.TURN_OFF)
    basic.pause(2000)
    maqueen.motor_stop(maqueen.Motors.ALL)

basic.forever(on_forever)
```

9.2.2. BLOQUES MAKECODE

```
[para siempre]
 ├ [reproduce secuencia (Melodía personalizada, Tempo 120 BPM, Modo hasta que termine)]
 ├ [establecer LED (Izquierdo, ENCENDIDO)]
 ├ [establecer LED (Derecho, ENCENDIDO)]
 ├ [pausa (ms): 2000]
 ├ [establecer LED (Izquierdo, APAGADO)]
 ├ [establecer LED (Derecho, APAGADO)]
 ├ [motor Run (Todos, Adelante, Velocidad 50)]
 ├ [pausa (ms): 1000]
 └ [motor Stop (Todos)]
```

1. Usar un bloque "para siempre".
2. Reproducir un sonido.
3. Encender ambos LEDs.
4. Esperar 500 ms.
5. Apagar los LEDs.
6. Avanzar a velocidad moderada.
7. Detenerse.

9.3. Solución para Ejercicio 3: Trompos con Maqueen

9.3.1. MICROPYTHON

```python
def on_forever():
    # Primera secuencia de movimiento
    maqueen.motor_run(maqueen.Motors.M1, maqueen.Dir.CW, 50)
    maqueen.motor_run(maqueen.Motors.M2, maqueen.Dir.CCW, 50)
    basic.pause(1000)

    # Primera parada de los motores
    maqueen.motor_stop(maqueen.Motors.ALL)
    basic.pause(500)
```

```python
    # Segunda secuencia de movimiento
    maqueen.motor_run(maqueen.Motors.M1, maqueen.Dir.CCW, 50)
    maqueen.motor_run(maqueen.Motors.M2, maqueen.Dir.CW, 50)
    basic.pause(1000)

    # Segunda parada de los motores
    maqueen.motor_stop(maqueen.Motors.ALL)
    basic.pause(500)

basic.forever(on_forever)
```

9.3.2. BLOQUES MAKECODE

```plaintext
[para siempre]
    ├─ [Motor: M1, sentido avanzar, velocidad 50]
    ├─ [Motor: M2, sentido retroceder, velocidad 50]
    ├─ [pausa (ms): 1000]
    ├─ [Parar motor: ambos]
    ├─ [pausa (ms): 500]
    ├─ [Motor: M1, sentido retroceder, velocidad 50]
    ├─ [Motor: M2, sentido avanzar, velocidad 50]
    ├─ [pausa (ms): 1000]
    ├─ [Parar motor: ambos]
    └─ [pausa (ms): 500]
```

1. Usar un bloque "para siempre".
2. Configurar el motor izquierdo para avanzar y el derecho para retroceder.
3. Esperar 1 segundo.

4. Detenerse.
5. Configurar el motor izquierdo para retroceder y el derecho para avanzar.
6. Esperar 1 segundo.
7. Detenerse.

9.4. Solución para Ejercicio 4: Evita obstáculos

9.4.1. MICROPYTHON

python Copiar código

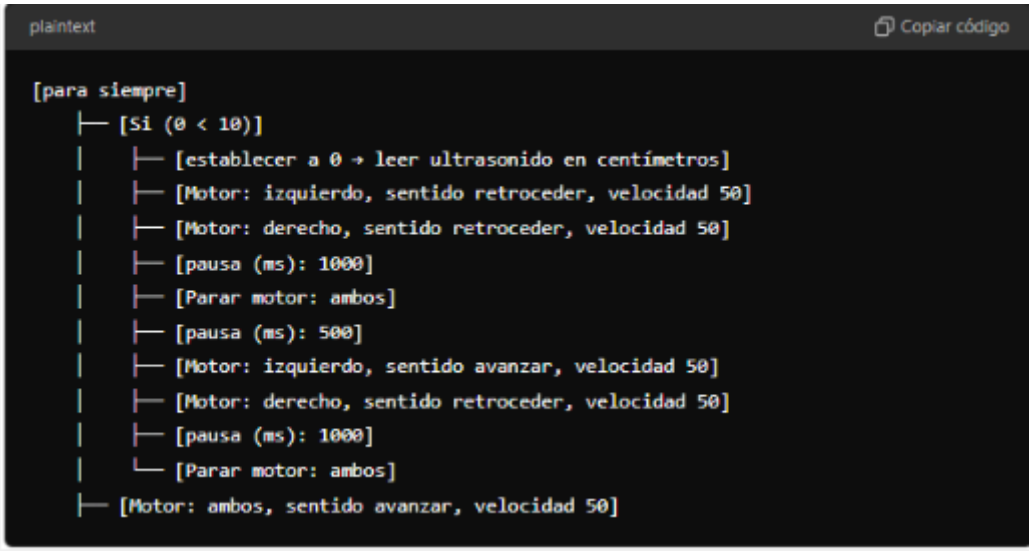
```python
def on_forever():
    # Medir la distancia con el sensor ultrasónico
    if maqueen.ultrasonic(PingUnit.CENTIMETERS) < 10:
        # Si hay un obstáculo cerca, retrocede
        maqueen.motor_run(maqueen.Motors.M1, maqueen.Dir.CCW, 50)  # Retrocede el motor iz
        maqueen.motor_run(maqueen.Motors.M2, maqueen.Dir.CCW, 50)  # Retrocede el motor de
        basic.pause(1000)  # Pausar durante 1 segundo

        # Detener ambos motores
        maqueen.motor_stop(maqueen.Motors.ALL)
        basic.pause(500)  # Pausar durante 0.5 segundos

        # Girar hacia la derecha para esquivar el obstáculo
        maqueen.motor_run(maqueen.Motors.M1, maqueen.Dir.CW, 50)   # Avanza el motor izqui
        maqueen.motor_run(maqueen.Motors.M2, maqueen.Dir.CCW, 50)  # Retrocede el motor de
        basic.pause(1000)  # Pausar durante 1 segundo
        # Detener ambos motores
        maqueen.motor_stop(maqueen.Motors.ALL)
        basic.pause(500)  # Pausar durante 0.5 segundos
    else:
        # Si no hay obstáculo, avanzar
        maqueen.motor_run(maqueen.Motors.ALL, maqueen.Dir.CW, 50)  # Ambos motores avanzan

# Ejecutar la función on_forever de forma continua
basic.forever(on_forever)
```

9.4.2. BLOQUES MAKECODE



1. Usar un bloque "para siempre".
2. Avanzar hacia adelante.
3. Si el sensor ultrasónico detecta un obstáculo a menos de 10 cm, retroceder y girar.
4. Continuar avanzando.

9.5. Solución para Ejercicio 5: Sigue líneas

9.5.1. MICROPYTHON

```python
def on_forever():
    if maqueen.read_patrol(maqueen.Patrol.PATROL_LEFT) == 0 and maqueen.read_patrol(maquee
        maqueen.motor_run(maqueen.Motors.ALL, maqueen.Dir.CW, 175)
    elif maqueen.read_patrol(maqueen.Patrol.PATROL_LEFT) == 1 and maqueen.read_patrol(maqu
        maqueen.motor_run(maqueen.Motors.M1, maqueen.Dir.CW, 175)
        maqueen.motor_run(maqueen.Motors.M2, maqueen.Dir.CW, 0)
    elif maqueen.read_patrol(maqueen.Patrol.PATROL_LEFT) == 0 and maqueen.read_patrol(maqu
        maqueen.motor_run(maqueen.Motors.M2, maqueen.Dir.CW, 175)
        maqueen.motor_run(maqueen.Motors.M1, maqueen.Dir.CW, 0)

basic.forever(on_forever)
```

9.5.2. BLOQUES MAKECODE

```plaintext
[para siempre]
    ├─ [Si (leer siguelínea izquierda == 0 y leer siguelínea derecha == 0)]
    │    ├─ [Motor: ambos, sentido avanzar, velocidad 175]
    ├─ [Sino Si (leer siguelínea izquierda == 1 y leer siguelínea derecha == 0)]
    │    ├─ [Motor: izquierdo, sentido avanzar, velocidad 175]
    │    ├─ [Motor: derecho, sentido avanzar, velocidad 0]
    ├─ [Sino Si (leer siguelínea derecha == 1 y leer siguelínea izquierda == 0)]
    │    ├─ [Motor: derecho, sentido avanzar, velocidad 175]
    │    ├─ [Motor: izquierdo, sentido avanzar, velocidad 0]
```

1. Usar un bloque "para siempre".
2. Verificar el estado de los sensores de siguelíneas.
3. Ajustar los motores para mantenerse sobre la línea.
4. Pausar 100 ms.

9.6. Solución para Ejercicio 6: Pilla - Pilla

9.6.1. MICROPYTHON

```python
derecha = Math.random_boolean()

def on_forever():
    if maqueen.ultrasonic(PingUnit.CENTIMETERS) < 20 and maqueen.ultrasonic(PingUnit.CENTI
        maqueen.motor_run(maqueen.Motors.ALL, maqueen.Dir.CW, 200)
    elif derecha == True:
        maqueen.motor_run(maqueen.Motors.M1, maqueen.Dir.CW, 200)
        maqueen.motor_run(maqueen.Motors.M2, maqueen.Dir.CW, 0)
    else:
        maqueen.motor_run(maqueen.Motors.M1, maqueen.Dir.CW, 0)
        maqueen.motor_run(maqueen.Motors.M2, maqueen.Dir.CW, 200)

basic.forever(on_forever)
```

9.6.2. BLOQUES MAKECODE

```plaintext
[establecer derecha a Math.randomBoolean()]
[para siempre]
    ├─ [Si (distancia < 20)]
    │    ├─ [Motor: ambos, sentido avanzar, velocidad 200]
    ├─ [Sino Si (derecha == verdadero)]
    │    ├─ [Motor: izquierdo, sentido avanzar, velocidad 200]
    │    ├─ [Motor: derecho, sentido detener, velocidad 0]
    └─ [Sino]
         ├─ [Motor: derecho, sentido avanzar, velocidad 200]
         ├─ [Motor: izquierdo, sentido detener, velocidad 0]
```

1. Usar un bloque "para siempre".
2. Si el sensor ultrasónico detecta un objeto a menos de 20 cm, avanzar hacia él.
3. Si no detecta ningún objeto, girar en una dirección aleatoria.
4. Pausar 100 ms.

ANEXOS

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