LOGICAL CODE EXPLANATION

Variable Declarations

```
const int trigPin = 9;

const int echoPin = 10;

const int buzzerPin = 11;
```

- **trigPin**: Assigned to pin 9, which is the trigger pin of the ultrasonic sensor. This pin will send an ultrasonic pulse.
- **echoPin**: Assigned to pin 10, which is the echo pin of the ultrasonic sensor. This pin will receive the reflected ultrasonic pulse.
- **buzzerPin**: Assigned to pin 11, connected to the buzzer. The buzzer will alert when an object is detected within a specific distance.

---

setup() Function

```
void setup()

{

  Serial.begin(9600);

  pinMode(trigPin, OUTPUT);

  pinMode(echoPin, INPUT);

  pinMode(buzzerPin, OUTPUT);

  Serial.println("Blind Stick is Ready");

}
```

1. **Serial.begin(9600)**: Initializes serial communication at a baud rate of 9600. This is used for debugging and displaying distance values in the Serial Monitor.
2. **pinMode(trigPin, OUTPUT)**: Sets the trigPin as an output, used to send the ultrasonic pulses.
3. **pinMode(echoPin, INPUT)**: Sets the echoPin as an input to read the reflected pulse.
4. **pinMode(buzzerPin, OUTPUT)**: Sets the buzzerPin as an output to control the buzzer.

5. **Serial.println("Blind Stick is Ready")**: Displays a message in the Serial Monitor indicating the system is ready.

---

loop() Function

```
void loop()
{
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
```

1. **digitalWrite(trigPin, LOW)**: Ensures the trigPin starts at a LOW state.
2. **delayMicroseconds(2)**: A short delay of 2 microseconds to stabilize the sensor.
3. **digitalWrite(trigPin, HIGH)**: Sends a HIGH signal to the trigPin for 10 microseconds, which triggers the ultrasonic sensor to emit an ultrasonic pulse.
4. **delayMicroseconds(10)**: The duration for which the trigPin stays HIGH to send the ultrasonic pulse.
5. **digitalWrite(trigPin, LOW)**: Sets the trigPin back to LOW after sending the pulse.

```
  long duration = pulseIn(echoPin, HIGH);
  long distance = duration * 0.034 / 2;
```

6. **pulseIn(echoPin, HIGH)**: Measures the time (in microseconds) it takes for the echo signal to return after the ultrasonic pulse hits an object.
7. **distance = duration * 0.034 / 2**: Converts the duration into distance in centimeters.
    - **0.034**: Speed of sound in cm/µs (343 m/s).
    - **/ 2**: Divides by 2 to account for the pulse traveling to the object and back.

```
Serial.print("Distance: ");
Serial.print(distance);
Serial.println(" cm");
```

8. **Serial.print()**: Sends the measured distance to the Serial Monitor for debugging or observation.

```
if (distance > 0 && distance <= 50)
{
  tone(buzzerPin, 1000);
}
else
{
  noTone(buzzerPin);
}
```

9. **Condition (if (distance > 0 && distance <= 50))**: Checks if the distance is between 0 and 50 cm:
    - **tone(buzzerPin, 1000)**: Activates the buzzer at 1000 Hz if an object is detected within 50 cm.
    - **noTone(buzzerPin)**: Turns off the buzzer if no object is detected within 50 cm.

```
delay(100);
```

10. **delay(100)**: Waits for 100 milliseconds before repeating the loop to reduce sensor updates frequency and stabilize readings.

Summary

1. Sends an ultrasonic pulse using trigPin.
2. Measures the time taken for the echo to return using echoPin.
3. Calculates the distance to the object based on the time.
4. Activates the buzzer if an object is within 50 cm; otherwise, the buzzer remains silent.
5. Repeats every 100 milliseconds.