IR –BUZZER

```
const int irPin = 10;     // Pin connected to the IR sensor output

const int buzzerPin = 9;   // Pin connected to the buzzer


void setup() {


  Serial.begin(9600);    // Initialize the serial monitor ... 9600 is communication speed


  pinMode(buzzerPin, OUTPUT);    // Set the buzzer pin as OUTPUT


  pinMode(irPin, INPUT);     // Set the IR sensor pin as INPUT

}


void loop() {


  int irValue = digitalRead(irPin);     // Read the value from the IR sensor


  Serial.println(irValue);      // Print the IR sensor value to the serial monitor


  // If the IR sensor detects an object (HIGH), turn on the buzzer

  if (irValue == HIGH) {

    digitalWrite(buzzerPin, HIGH); // Turn buzzer on

  }

  else {

    digitalWrite(buzzerPin, LOW); // Turn buzzer off

  }


  // Small delay for stability

  delay(100);

}
```

IR-LED

```
const int irPin = 10;   // Pin connected to the IR sensor output

const int ledPin = 9;   // Pin connected to the LED

void setup() {

 Serial.begin(9600);    // Initialize the serial monitor

 pinMode(ledPin, OUTPUT);   // Set the LED pin as OUTPUT

 pinMode(irPin, INPUT);     // Set the IR pin as INPUT

}

void loop() {

 int irValue = digitalRead(irPin);  // Read the value from the IR sensor

 Serial.println(irValue);    // Print the value to the serial monitor

 if (irValue == HIGH) {     // If the IR sensor detects a signal (HIGH), turn on the LED

 digitalWrite(ledPin, HIGH); // Turn LED on

 } else {

 digitalWrite(ledPin, LOW); // Turn LED off

 }

 // Small delay for stability

 delay(100);

}
```

EXPLAINATION

This Arduino code is used to detect an object or obstacle using an infrared (IR) sensor and control an LED based on the sensor's reading. The LED lights up whenever the IR sensor detects an object or signal.

Code Explanation:

Variable Definitions:

irPin: Sets pin 10 for the IR sensor, which detects objects or obstacles.

ledPin: Sets pin 9 for the LED, which will turn on or off depending on the IR sensor reading.

setup function:

Serial.begin(9600);: Starts the Serial Monitor for viewing data sent from the Arduino, with a communication speed of 9600 baud.

pinMode(ledPin, OUTPUT);: Configures ledPin (pin 9) as an output to control the LED.

pinMode(irPin, INPUT);: Configures irPin (pin 10) as an input to read the IR sensor signal.

loop function:

int irValue = digitalRead(irPin);: Reads the digital signal from the IR sensor and stores it in irValue. HIGH means an object is detected, while LOW means no object is detected.

Serial.println(irValue);: Prints the irValue to the Serial Monitor for monitoring sensor output in real-time.

Conditional check:

If irValue is HIGH (indicating detection), the LED is turned on using digitalWrite(ledPin, HIGH);.

If irValue is LOW, the LED is turned off using digitalWrite(ledPin, LOW);.

delay(100);: Adds a short 100 ms delay to prevent excessive reading, which can provide more stable readings.

TEMPERATURE-BUZZER

```
#include <dht.h>

#define inPin 2     // Pin number for DHT11 data

#define buzzerPin 10    // Pin number for the buzzer

dht DHT;

const float tempThreshold = 37.0;   // Temperature threshold in Celsius

const float humidityThreshold = 70.0;   // Humidity threshold in percentage


void setup() {

 Serial.begin(9600);

 pinMode(buzzerPin, OUTPUT);       // Set buzzer pin as output

 digitalWrite(buzzerPin, LOW);      // Ensure the buzzer is off initially

}


void loop() {

 int readData = DHT.read11(inPin);

 float t = DHT.temperature;     // Read temperature

 float h = DHT.humidity;        // Read humidity

 Serial.print("Temperature = ");

 Serial.print(t);

 Serial.print("°C | ");

 Serial.print((t * 9.0) / 5.0 + 32.0);       // Convert Celsius to Fahrenheit

 Serial.println("°F ");


 Serial.print("Humidity = ");

 Serial.print(h);

 Serial.println("% ");


 if (t > tempThreshold || h > humidityThreshold) {      // Check if temperature or humidity exceeds thresholds

 tone(buzzerPin, 1000); // Activate buzzer at 1000 Hz

 delay(200); // Sound for 200 milliseconds

 noTone(buzzerPin); // Turn off buzzer

 Serial.println("Alert! Threshold exceeded!");

 }

 delay(1000);       // Wait four seconds before next reading

}
```

EXPLAINATION:

This Arduino code is used for temperature and humidity monitoring using a DHT11 sensor. It sounds a buzzer alarm when temperature or humidity readings exceed defined thresholds (37°C for temperature and 70% for humidity). This setup is useful for creating simple environmental monitoring systems, such as in a room or greenhouse, to alert users when the conditions go beyond safe levels.

Code Explanation

Libraries and Pin Setup:

#include <dht.h>: Includes the library for the DHT11 sensor, which simplifies reading temperature and humidity values.

#define inPin 2: Sets pin 2 as the data pin for the DHT11 sensor.

#define buzzerPin 10: Sets pin 10 as the output pin for the buzzer.

Thresholds:

const float tempThreshold = 37.0;: Defines the temperature threshold (37°C).

const float humidityThreshold = 70.0;: Defines the humidity threshold (70%).

setup function:

Serial.begin(9600);: Initializes the Serial Monitor at a baud rate of 9600 for output.

pinMode(buzzerPin, OUTPUT);: Sets the buzzerPin as an output for controlling the buzzer.

digitalWrite(buzzerPin, LOW);: Ensures the buzzer is off initially.

loop function:

Reading Temperature and Humidity:

int readData = DHT.read11(inPin);: Reads data from the DHT11 sensor.

float t = DHT.temperature;: Retrieves the temperature in Celsius from the sensor.

float h = DHT.humidity;: Retrieves the humidity in percentage.

Displaying Values:

Serial.print("Temperature = "); and subsequent lines print the temperature in Celsius and Fahrenheit.

Serial.print("Humidity = "); and the following lines print the humidity percentage.

Threshold Check:

if (t > tempThreshold || h > humidityThreshold): Checks if the temperature exceeds 37°C or if humidity exceeds 70%.

tone(buzzerPin, 1000);: Activates the buzzer at a frequency of 1000 Hz.

delay(200);: Sounds the buzzer for 200 milliseconds.

noTone(buzzerPin);: Turns off the buzzer after the alert.

Serial.println("Alert! Threshold exceeded!");: Prints an alert message to the Serial Monitor.

Delay:

delay(1000);: Waits for 1 second before taking the next reading to avoid rapid polling of the sensor.

TEMPERATURE-LED

```
#include <dht.h>

#define inPin 2          // Pin number for DHT11 data

#define ledPin 10        // Pin number for the LED

dht DHT;

const float tempThreshold = 37.0;      // Temperature threshold in Celsius

const float humidityThreshold = 70.0;   // Humidity threshold in percentage


void setup() {

  Serial.begin(9600);

  pinMode(ledPin, OUTPUT);         // Set LED pin as output

  digitalWrite(ledPin, LOW);       // Ensure the LED is off initially

}


void loop() {

  int readData = DHT.read11(inPin);

  float t = DHT.temperature;       // Read temperature

  float h = DHT.humidity;          // Read humidity


  // Display temperature and humidity values in Serial Monitor

  Serial.print("Temperature = ");

  Serial.print(t);

  Serial.print("°C | ");

  Serial.print((t * 9.0) / 5.0 + 32.0); // Convert Celsius to Fahrenheit

  Serial.println("°F");


  Serial.print("Humidity = ");

  Serial.print(h);

  Serial.println("% ");


  // Check if temperature or humidity exceeds thresholds
  if (t > tempThreshold || h > humidityThreshold) {
    digitalWrite(ledPin, HIGH);     // Turn LED on
    Serial.println("Alert! Threshold exceeded!");
  } else {
    digitalWrite(ledPin, LOW);      // Turn LED off
  }

  delay(1000);                      // Wait one second before the next reading
}
```

EXPLAINATION :

Code Explanation

Pin Setup:

#define ledPin 10: Uses pin 10 to control the LED.

Thresholds:

tempThreshold and humidityThreshold define the temperature (37°C) and humidity (70%) levels to trigger the alert.

Setup Function:

pinMode(ledPin, OUTPUT);: Sets ledPin as an output for controlling the LED.

digitalWrite(ledPin, LOW);: Keeps the LED off initially.

Loop Function:

Reads temperature and humidity from the DHT11 sensor.

Displays the readings on the Serial Monitor.

Alert Condition:

If either temperature or humidity exceeds the threshold, the LED lights up (digitalWrite(ledPin, HIGH);) and an alert message is printed.

If thresholds are within safe limits, the LED turns off (digitalWrite(ledPin, LOW);).