

```
//Raspberry Pi interface with LED control - Program 1
```

```
import RPi.GPIO as GPIO
```

```
import time
```

```
# Disable warnings (e.g., "GPIO already in use")
```

```
GPIO.setwarnings(False)
```

```
# Use BCM pin numbering
```

```
GPIO.setmode(GPIO.BCM)
```

```
# Set GPIO 17 as output
```

```
LED_PIN = 17
```

```
GPIO.setup(LED_PIN, GPIO.OUT)
```

```
try:
```

```
while True:
```

```
    GPIO.output(LED_PIN, True) # LED ON
```

```
    time.sleep(1) # 1 second delay
```

```
    GPIO.output(LED_PIN, False) # LED OFF
```

```
    time.sleep(1) # 1 second delay
```

```
except KeyboardInterrupt:
```

```
    # Gracefully clean up on Ctrl+C
```

```
    GPIO.cleanup()
```

```
//IOT Data Processing with Raspberry pi pico W + HC-SR04 Ultrasonic Sensor + ThingSpeak - Program
```

```
import network
```

```
import urequests
```

```
import time
```

```
import machine
```

```
ssid = "Wokwi-GUEST"
```

```
password = ""
```

```
server = "api.thingspeak.com"
```

```
channel_id = 1234567
```

```
write_api_key = "XXXXXXXXXXXXXXXXXX"
```

```
trigger_pin = machine.Pin(15, machine.Pin.OUT)
```

```
echo_pin = machine.Pin(14, machine.Pin.IN)
```

```
led_alert_pin = machine.Pin(13, machine.Pin.OUT)
```

```
DISTANCE_CLOSE_THRESHOLD = 10.0
```

```
def connect_wifi():
```

```
    wlan = network.WLAN(network.STA_IF)
```

```
    wlan.active(True)
```

```
    wlan.connect(ssid, password)
```

```

print("Connecting to WiFi...")
while not wlan.isconnected():
    time.sleep(0.5)
    print(".")
print("WiFi connected")
print("IP address:", wlan.ifconfig()[0])
def measure_distance():
    trigger_pin.off()
    time.sleep_us(2)
    trigger_pin.on()
    time.sleep_us(10)
    trigger_pin.off()
    while echo_pin.value() == 0:
        signal_off = time.ticks_us()
    while echo_pin.value() == 1:
        signal_on = time.ticks_us()
    time_passed = signal_on - signal_off
    distance = (time_passed * 0.0343) / 2
    return distance
connect_wifi()
last_send = 0
while True:
    if time.time() - last_send < 20:
        time.sleep(1)
        continue
    last_send = time.time()
    distance = measure_distance()
    print("Distance:", round(distance, 1), "cm")
    alert = 0
    alert_msg = "Normal"
    led_alert_pin.off()
    if distance < DISTANCE_CLOSE_THRESHOLD:
        alert = 1
        alert_msg = "OBJECT TOO CLOSE"
        led_alert_pin.on()
    print("Alert:", alert, alert_msg)
    url = "https://{}/update?api_key={}&field1={:.1f}&field2={}".format(

```

```

        server, write_api_key, distance, alert}

try:
    response = urequests.get(url)
    if response.status_code == 200:
        print("Data sent successfully!")
    response.close()
except Exception as e:
    print("HTTP request failed:", e)
time.sleep(1)

//Raspberry Pi Interface with IR (Obstacle) Sensor - program 1
import RPi.GPIO as GPIO

import time

GPIO.setmode(GPIO.BCM)
GPIO.setup(18, GPIO.IN)

print("IR Sensor Test - Press Ctrl+C to stop")

try:
    while True:
        if GPIO.input(18) == GPIO.LOW: # LOW when object is close
            print("Object Detected")
        else:
            print("No Object")
        time.sleep(0.5)
except KeyboardInterrupt:
    GPIO.cleanup()

//IoT Data Processing with ESP32 + DHT22 +ThingSpeak - program 2
#include <WiFi.h>
#include <HTTPClient.h>
#include <WiFiClientSecure.h>
#include <DHTesp.h>

const char* ssid = "Wokwi-GUEST";
const char* password = "";
const char* server = "api.thingspeak.com";
const int channelID = 1234567;
const char* writeAPIKey = "XXXXXXXXXXXXXXXXXX";

#define DHT_PIN 15
#define LED_ALERT_PIN 2

```

```

#define TEMP_HIGH_THRESHOLD 30.0

#define TEMP_LOW_THRESHOLD 18.0

#define HUM_HIGH_THRESHOLD 80.0

DHTesp dht;

WiFiClientSecure client;

void setup() {
    Serial.begin(115200);

    pinMode(LED_ALERT_PIN, OUTPUT);
    digitalWrite(LED_ALERT_PIN, LOW);

    dht.setup(DHT_PIN, DHTesp::DHT22);

    WiFi.begin(ssid, password);

    Serial.print("Connecting to WiFi...");

    while (WiFi.status() != WL_CONNECTED) {
        delay(500);
        Serial.print(".");
    }

    Serial.println("\nWiFi connected");

    Serial.print("IP address: ");
    Serial.println(WiFi.localIP());

    client.setInsecure();
}

void loop() {
    static unsigned long lastSend = 0;

    if (millis() - lastSend < 20000) {
        delay(100);
        return;
    }

    lastSend = millis();

    TempAndHumidity data = dht.getTempAndHumidity();

    if (dht.getStatus() != DHTesp::ERROR_NONE) {
        Serial.println("DHT read error!");
        return;
    }

    float temp = data.temperature;

    float hum = data.humidity;

    Serial.printf("Temp: %.1f °C | Humidity: %.1f %%\n", temp, hum);

    int alert = 0;

```

```

String alertMsg = "Normal";

digitalWrite(LED_ALERT_PIN, LOW);

if (temp > TEMP_HIGH_THRESHOLD) {

    alert = 1;

    alertMsg = "HIGH TEMP";

    digitalWrite(LED_ALERT_PIN, HIGH);

}

else if (temp < TEMP_LOW_THRESHOLD) {

    alert = 1;

    alertMsg = "LOW TEMP";

    digitalWrite(LED_ALERT_PIN, HIGH);

}

else if (hum > HUM_HIGH_THRESHOLD) {

    alert = 1;

    alertMsg = "HIGH HUMIDITY";

    digitalWrite(LED_ALERT_PIN, HIGH);

}

Serial.println("Alert: " + alertMsg);

if (WiFi.status() == WL_CONNECTED) {

    HTTPClient http;

    String url = "https://" + String(server) +

        "/update?api_key=" + String(writeAPIKey) +

        "&field1=" + String(temp, 1) +

        "&field2=" + String(hum, 1) +

        "&field3=" + String(alert);

    http.begin(client, url);

    int httpCode = http.GET();

    if (httpCode > 0) {

        Serial.printf("ThingSpeak Response: %d\n", httpCode);

        if (httpCode == 200) {

            Serial.println("Data sent successfully!");

        }

    } else {

        Serial.println("HTTP request failed");

    }

    http.end();

} else {

```

```
        Serial.println("WiFi lost");
    }
    delay(100);
}
```

//Raspberry Pi Interface with Ultrasonic Sensor - Program 1

```
import RPi.GPIO as GPIO

import time

GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)

TRIG = 20
ECHO = 21

GPIO.setup(TRIG, GPIO.OUT)
GPIO.setup(ECHO, GPIO.IN)

try:
    print("Press Ctrl+C to stop\n")
    while True:
        # Send trigger pulse
        GPIO.output(TRIG, False)

        time.sleep(0.0002)

        GPIO.output(TRIG, True)
        time.sleep(0.00001)
        GPIO.output(TRIG, False)

        # Capture echo times
        while GPIO.input(ECHO) == 0:
            start = time.time()

        while GPIO.input(ECHO) == 1:
            end = time.time()

        # Calculate distance in cm
        distance = (end - start) * 34300 / 2

        # Use .format() for printing
        print("Distance = {:.1f} cm".format(distance))

        time.sleep(0.5)

except KeyboardInterrupt:
    print("\nMeasurement stopped by user.")

GPIO.cleanup()
```

//Raspberry Pi Interface with DHT11 Temperature & Humidity Sensor - Program 1

```
import RPi.GPIO as GPIO

import dht11

import time

# GPIO setup

GPIO.setwarnings(False)

GPIO.setmode(GPIO.BCM)

GPIO.cleanup()

# Setup sensor

instance = dht11.DHT11(pin=21) # GPIO21

while True:

    result = instance.read()

    if result.is_valid():

        print("Temperature: {} C Humidity: {} %".format(result.temperature, result.humidity))

    else:

        print("Waiting for valid data...")

    time.sleep(2)
```

//Ultrasonic Sensor and Relay Interface with Raspberry Pi - Program 1

```
import RPi.GPIO as GPIO

import time

GPIO.setmode(GPIO.BCM)

TRIG = 23

ECHO = 24

RELAY = 18

GPIO.setup(TRIG, GPIO.OUT)

GPIO.setup(ECHO, GPIO.IN)

GPIO.setup(RELAY, GPIO.OUT)

try:

    while True:

        GPIO.output(TRIG, False)

        time.sleep(0.05)

        GPIO.output(TRIG, True)

        time.sleep(0.00001)

        GPIO.output(TRIG, False)

        while GPIO.input(ECHO) == 0:

            start = time.time()
```

```

while GPIO.input(ECHO) == 1:
    end = time.time()
    distance = (end - start) * 17150
    print(f"Distance: {distance:.1f} cm")
    GPIO.output(RELAY, GPIO.HIGH if distance < 20 else GPIO.LOW)
except KeyboardInterrupt:
    GPIO.cleanup()

```

//IR Sensor and Relay Interface with Raspberry Pi - Program 1

```

import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BCM)

IR_PIN = 17
BUZZER = 27

GPIO.setup(IR_PIN, GPIO.IN)
GPIO.setup(BUZZER, GPIO.OUT)

try:
    while True:
        if GPIO.input(IR_PIN) == GPIO.LOW: # Object detected
            GPIO.output(BUZZER, GPIO.HIGH)
        else:
            GPIO.output(BUZZER, GPIO.LOW)
        time.sleep(0.1)
except KeyboardInterrupt:
    GPIO.cleanup()

```

//DHT11 Sensor and Relay Interface with Raspberry Pi - Program 1

```

import Adafruit_DHT
import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BCM)

RELAY = 20

GPIO.setup(RELAY, GPIO.OUT)

sensor = Adafruit_DHT.DHT11

pin = 21

try:
    while True:

```

```
humidity, temperature = Adafruit_DHT.read_retry(sensor, pin)

if humidity is not None and temperature is not None:

    print(f"Temp={temperature:.1f}°C Humidity={humidity:.1f}%")

    GPIO.output(RELAY, GPIO.HIGH if temperature > 30 else GPIO.LOW)

else:

    print("Sensor failure. Check wiring.")

    time.sleep(2)

except KeyboardInterrupt:

    GPIO.cleanup()
```