PROGRAM 4

Board & Environment Setup (Raspberry Pi 4)

Board: Raspberry Pi 4
OS: Raspberry Pi OS
Language: Python 3 (ONLY)

Required Python Modules / Packages (Install BEFORE running program)

Run the following commands in the terminal:

```
# Update system (recommended)
sudo apt update

# Install HTTP library for cloud communication
pip3 install requests

# Install Adafruit Blinka (required for board module)
pip3 install adafruit-blinka

# Install DHT sensor library (Python 3 compatible)
pip3 install adafruit-circuitpython-dht
```

> **Note:** Do NOT install `Adafruit_DHT`. It is a legacy Python-2-based library.

Program Title

Environmental Data Collection and Upload to Cloud using Raspberry Pi and ThingSpeak

---

Aim

To collect temperature and humidity data using Raspberry Pi and upload the data to a ThingSpeak cloud platform for remote monitoring.

Components Used

- Raspberry Pi 4
- DHT11 Temperature and Humidity Sensor (3-pin module)

- Jumper wires
- Internet connection (Wi-Fi / Ethernet)
- Power supply

Pin Configuration

DHT11 Sensor Wiring

DHT11 Pin	Raspberry Pi Physical Pin	GPIO / Board Name
VCC	Pin 2 / Pin 4	5V
DATA	Pin 7	GPIO4 / `board.D4`
GND	Pin 6	GND

Important: The DHT11 module is powered using **5V** for stable operation.

ThingSpeak API Key Creation (Web Steps)

1. Open **ThingSpeak** website and log in using a MathWorks account
2. Go to **Channels → My Channels → New Channel**
3. Enable the following fields:
4. **Field 1** → Temperature
5. **Field 2** → Humidity
6. Click **Save Channel**
7. Open the created channel and go to the **API Keys** tab
8. Copy the **Write API Key** (used for uploading data)

> **Note:** Field 1 is mapped to temperature and Field 2 is mapped to humidity in the program.

Program Code (Python 3 – Final Version)

```
# ------------------------------------------------------------
# PROGRAM 4: Upload Temperature and Humidity Data to ThingSpeak
# Platform : Raspberry Pi 4
# Language : Python 3
# Sensor   : DHT11 (3-pin module, powered with 5V)
# ------------------------------------------------------------

import time                    # For delay and timing control
import board                   # For GPIO pin abstraction
import adafruit_dht            # For DHT11 sensor communication
```

```
import requests                # For HTTP communication with ThingSpeak

# Initialize DHT11 sensor on GPIO4 (Physical Pin 7)
dht = adafruit_dht.DHT11(board.D4)

# ThingSpeak configuration
API_KEY = "YOUR_WRITE_API_KEY"   # Replace with your Write API Key
THINGSPEAK_URL = "https://api.thingspeak.com/update"

# Continuous data upload loop
while True:
    try:
        # Read temperature and humidity from sensor
        temperature = dht.temperature      # Temperature in Celsius
        humidity = dht.humidity            # Humidity in percentage

        # Prepare data payload for ThingSpeak
        payload = {
            'api_key': API_KEY,
            'field1': temperature,   # Field 1 -> Temperature
            'field2': humidity       # Field 2 -> Humidity
        }

        # Send data to ThingSpeak using HTTP GET request
        response = requests.get(THINGSPEAK_URL, params=payload)

        # Print confirmation
        print("Uploaded -> Temperature:", temperature,
              "Humidity:", humidity)

        # ThingSpeak minimum update interval is 15 seconds
        time.sleep(15)

    except RuntimeError as error:
        # Handles intermittent DHT sensor read errors
        print("Sensor error:", error)
        time.sleep(2)

    except Exception as error:
        # Handles internet or HTTP request errors
        print("Upload error:", error)
        time.sleep(5)
```

Output / Result

- Temperature and humidity data are uploaded to ThingSpeak cloud
- Real-time graphs for Field 1 (Temperature) and Field 2 (Humidity) are visible on the ThingSpeak dashboard

---

Possible Viva / Exam Questions

1. Why is ThingSpeak used in this program?
2. What is the purpose of the Write API Key?
3. Why is a 15-second delay required between uploads?
4. Which field is used for temperature and humidity?
5. What happens if internet connectivity is lost?

Conclusion

The program demonstrates IoT-based environmental monitoring by acquiring sensor data using Raspberry Pi and uploading it to a cloud platform for remote visualization and analysis.