PROGRAMMING THE ARDUINO FOR IOT

1. Components Needed

Component	Purpose
Arduino Uno/Nano	Base microcontroller
ESP8266 / NodeMCU	WiFi module for internet connection
Sensors (DHT11, etc.)	Collect temperature/humidity data
Actuators (Relay, Buzzer)	Perform actions
IoT Platform (Thingspeak, Blynk)	Cloud storage & display

2. Programming Arduino for IoT – Steps

Step 1: Setup Hardware

You can use either:

A. Arduino + WiFi Module (ESP8266)

- Connect ESP8266 to Arduino using serial pins

- Requires extra wiring

B. NodeMCU or ESP32 (Recommended)

- Built-in WiFi support
- No need for separate Arduino board

Step 2: Install Required Libraries

Use Arduino IDE and install:

- ESP8266WiFi.h or WiFi.h (for ESP32)
- ThingSpeak.h (for cloud)
- BlynkSimpleEsp8266.h (for Blynk app)

To install:

Sketch > Include Library > Manage Libraries → Search and install required ones.

Step 3: Connect to WiFi

Sample code to connect NodeMCU to WiFi:

```
#include <ESP8266WiFi.h>


const char* ssid = "YourWiFiSSID";
```

```
const char* password = "YourWiFiPassword";


void setup() {
  Serial.begin(9600);
  WiFi.begin(ssid, password);


  Serial.print("Connecting to WiFi");
  while (WiFi.status() != WL_CONNECTED) {
    delay(1000);
    Serial.print(".");
  }
  Serial.println("Connected!");
}


void loop() {
  // Add sensor or cloud code here
}
```

Step 4: Send Data to Cloud (Example: Thingspeak)

```
#include <ESP8266WiFi.h>
#include "ThingSpeak.h"

const char* ssid = "YourWiFiSSID";
const char* password = "YourPassword";
WiFiClient client;

unsigned long channelID = 123456;      // Your ThingSpeak Channel ID
const char* writeAPIKey = "ABC123XYZ"; // Your Write API Key

void setup() {
  Serial.begin(9600);
  WiFi.begin(ssid, password);
```

```
  ThingSpeak.begin(client);
}

void loop() {
  int value = analogRead(A0); // Read data (e.g., from sensor)
  ThingSpeak.writeField(channelID, 1, value, writeAPIKey);
  delay(15000); // Update every 15 sec
}
```

Step 5: Display or Control via Mobile App (Example: Blynk)

1. Download Blynk app
2. Create a new project and get the auth token
3. Use it in code:

```
#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>

char auth[] = "YourBlynkToken";
```

```
char ssid[] = "YourWiFiSSID";

char pass[] = "YourPassword";


void setup() {

  Blynk.begin(auth, ssid, pass);

}


void loop() {

  Blynk.run();

}
```

Add widgets in the Blynk app to control or view sensor values.

3. IoT Projects Using Arduino

Project Idea	**Modules Used**
Smart Weather Station	DHT11 + NodeMCU + Thingspeak

Home Automation via Phone	NodeMCU + Relay + Blynk
Smart Door Lock	RFID + NodeMCU + Firebase
IoT-based Soil Moisture Monitor	Soil Sensor + ESP8266 + LCD Display

4. Platforms to Connect Arduino IoT Projects

Platform	Purpose	Free Tier?
Thingspeak	IoT data logging and graphs	Yes
Blynk	IoT mobile app for control	Yes
IFTTT	Automation and alerts	Yes
Firebase	Real-time database	Yes

Summary – Programming Arduino for IoT

Step	Description
Connect WiFi	Use ESP8266 or NodeMCU

Add Libraries	WiFi, Cloud (ThingSpeak, Blynk)
Read Sensors	Use analogRead() or digitalRead()
Upload to Cloud	Send values using APIs or SDKs
Use Apps/Graphs	View/control via phone or web dashboards

Basic Arduino Programs

1. Blink LED (Hello World of Arduino)

Purpose: Turns an LED on and off every second.

```
void setup() {
  pinMode(13, OUTPUT); // Set pin 13 as output
}


void loop() {
  digitalWrite(13, HIGH); // Turn LED ON
  delay(1000);            // Wait 1 second
```

```
  digitalWrite(13, LOW);  // Turn LED OFF
  delay(1000);            // Wait 1 second
}
```

2. LED with Button

Purpose: Turns on the LED when the button is pressed.

```
int button = 2;
int led = 13;

void setup() {
  pinMode(button, INPUT);
  pinMode(led, OUTPUT);
}

void loop() {
  if (digitalRead(button) == HIGH) {
    digitalWrite(led, HIGH);
```

```
  } else {
    digitalWrite(led, LOW);
  }
}
```

3. Analog Sensor Reading (e.g., LDR, Potentiometer)

Purpose: Reads analog value from pin A0.

```
void setup() {
  Serial.begin(9600);
}

void loop() {
  int sensorValue = analogRead(A0);
  Serial.println(sensorValue);
  delay(500);
}
```

4. Temperature Reading using DHT11 Sensor

```
#include <DHT.h>

#define DHTPIN 2

#define DHTTYPE DHT11

DHT dht(DHTPIN, DHTTYPE);


void setup() {
  Serial.begin(9600);
  dht.begin();
}


void loop() {
  float temp = dht.readTemperature();
  float hum = dht.readHumidity();


  Serial.print("Temp: ");
  Serial.print(temp);
```

```
  Serial.print(" C, Humidity: ");
  Serial.print(hum);
  Serial.println(" %");

  delay(2000);
}
```

5. Buzzer Alert

Purpose: Turns on buzzer for 1 second, then off.

```
int buzzer = 8;

void setup() {
  pinMode(buzzer, OUTPUT);
}

void loop() {
  digitalWrite(buzzer, HIGH);
```

```
  delay(1000);
  digitalWrite(buzzer, LOW);
  delay(1000);
}
```

6. Servo Motor Control

```
#include <Servo.h>
Servo myservo;

void setup() {
  myservo.attach(9); // Connect servo to pin 9
}

void loop() {
  myservo.write(0);
  delay(1000);
  myservo.write(90);
```

```
  delay(1000);
  myservo.write(180);
  delay(1000);
}
```