**Breakdown of the Code**

**1. Including Necessary Library**

```
#include <SoftwareSerial.h>
```

- The **SoftwareSerial** library is included to allow serial communication on digital pins other than the default **pins 0 (RX) and 1 (TX)**.
- However, in this code, the HC-05 Bluetooth module is incorrectly assigned to **pins 0 and 1**, which conflicts with the Arduino's built-in hardware serial. This should be changed to other pins like **10 and 11**.

---

**2. Defining Pin Assignments**

```
int sensorPin = 2;  // PIR sensor OUT pin
int relayPin = 3;   // Relay IN pin
int bluetoothTx = 0; // HC-05 TX (Connect to Arduino RX)
int bluetoothRx = 1; // HC-05 RX (Connect to Arduino TX)
```

- **sensorPin (PIR Sensor OUT - Pin 2)**: Reads motion detection status.
- **relayPin (Relay Module IN - Pin 3)**: Controls the light (turns ON/OFF the bulb).
- **bluetoothTx & bluetoothRx (Pins 0 & 1)**: These are assigned for HC-05 communication, but **this is incorrect because Pins 0 & 1 are used for USB communication with the PC.**
    - **Fix**: Change these to other digital pins (e.g., 10 and 11) and use SoftwareSerial.

---

**3. Defining State Variables**

```
bool isLightOn = false;   // Light state (false = OFF, true = ON)
bool motionDetected = false; // Tracks if motion is currently detected
```

- **isLightOn**: Keeps track of whether the bulb is ON (true) or OFF (false).
- **motionDetected**: Prevents repeated triggers when motion is detected continuously.

---

**4. Initializing Bluetooth Communication**

```
SoftwareSerial BT(bluetoothTx, bluetoothRx); // Create Bluetooth serial
```

- **Creates a software serial communication channel** for HC-05 Bluetooth. However, as mentioned earlier, using **pins 0 and 1 is incorrect** because they interfere with USB communication. Instead, use:

```
SoftwareSerial BT(10, 11); // Assign custom pins for Bluetooth TX/RX
```

---

**5. Setup Function**

```
void setup() {
  pinMode(sensorPin, INPUT);   // Set PIR sensor as input
  pinMode(relayPin, OUTPUT);   // Set relay as output
  digitalWrite(relayPin, LOW); // Ensure relay is OFF initially

  Serial.begin(9600);  // Initialize serial communication
  BT.begin(9600);      // Start Bluetooth communication
}
```

**Functionality:**

- **Sets up the PIR sensor** (sensorPin) as an input.
- **Configures the relay module** (relayPin) as an output.
- **Turns off the light initially** (digitalWrite(relayPin, LOW);).
- **Starts Serial Communication (USB & Bluetooth)**:
    - Serial.begin(9600); → For debugging via the **Serial Monitor**.
    - BT.begin(9600); → Starts Bluetooth communication at **9600 baud rate**.

---

**6. Loop Function (Main Execution)**

```
void loop() {
  int motionState = digitalRead(sensorPin); // Read PIR sensor state
```

- Reads the **PIR sensor state** (HIGH = motion detected, LOW = no motion).
- The variable motionState stores the result.

---

**7. Handling Motion Detection**

```
  if (motionState == HIGH && !motionDetected) {
    motionDetected = true; // Mark motion as detected
```

- If motion is **detected (HIGH)** and it was not previously detected (!motionDetected), update the motionDetected flag.

---

**8. Toggling Light Based on Motion**

```
    if (isLightOn) {
      Serial.println("Motion detected! Bulb OFF");
      BT.println("Bulb OFF (Motion)");
      digitalWrite(relayPin, LOW);
      isLightOn = false;
    } else {
      Serial.println("Motion detected! Bulb ON");
      BT.println("Bulb ON (Motion)");
      digitalWrite(relayPin, HIGH);
      isLightOn = true;
    }
```

- If the **light is already ON**, turn it **OFF** and send updates via Serial and Bluetooth.
- If the **light is OFF**, turn it **ON**.

---

**9. Reset Motion Detection**

```
if (motionState == LOW) {
  motionDetected = false; // Reset for next detection
}
```

- When no motion is detected, reset motionDetected so that the sensor can detect new motion events.

---

**10. Bluetooth Control (Override PIR Sensor)**

```
if (BT.available()) {
  char command = BT.read(); // Read incoming Bluetooth command
```

- Checks if there is an **incoming Bluetooth command** from a mobile app.

---

**11. Handling Bluetooth Commands**

```
  if (command == '1') {  // If '1' is received, turn ON light
    digitalWrite(relayPin, HIGH);
    Serial.println("Bulb ON via Bluetooth");
    BT.println("Bulb ON (Bluetooth)");
    isLightOn = true;
  }
  else if (command == '0') {  // If '0' is received, turn OFF light
    digitalWrite(relayPin, LOW);
    Serial.println("Bulb OFF via Bluetooth");
    BT.println("Bulb OFF (Bluetooth)");
    isLightOn = false;
  }
```

- **Receives commands** from the mobile app:
  - '1' → Turns the **light ON**.
  - '0' → Turns the **light OFF**.

- Sends feedback to **both the Serial Monitor** and **the Bluetooth app**.