```
#include <OneWire.h>
#include <DallasTemperature.h>

// Pin Definitions
#define ONE_WIRE_BUS 4         // DS18B20 data pin
#define HEATER_PIN 5           // Output pin to simulate heater (LED or virtual
heater)
#define LED_PIN 6              // Optional: Visual indicator

// Temperature thresholds (in °C)
#define START_HEATING 35.0
#define STOP_HEATING 65.0
#define OVERHEAT_TEMP 75.0

// Setup OneWire and Dallas Temperature
OneWire oneWire(ONE_WIRE_BUS);
DallasTemperature sensors(&oneWire);

// System States
enum HeaterState {
  IDLE,
  HEATING,
  STABILIZING,
  TARGET_REACHED,
  OVERHEAT
};

HeaterState currentState = IDLE;
float currentTemp = 0.0;

void setup() {
  Serial.begin(9600);
  sensors.begin();
  pinMode(HEATER_PIN, OUTPUT);
  pinMode(LED_PIN, OUTPUT);
  digitalWrite(HEATER_PIN, LOW);
  digitalWrite(LED_PIN, LOW);
  Serial.println("System Initialized: IDLE");
}

void loop() {
  sensors.requestTemperatures();
  currentTemp = sensors.getTempCByIndex(0);
  updateState(currentTemp);
  logStatus(currentTemp);

  delay(1000);  // Read every second
}

void updateState(float temp) {
  switch (currentState) {
    case IDLE:
      if (temp < START_HEATING) {
        currentState = HEATING;
        digitalWrite(HEATER_PIN, HIGH);
      }
      break;

    case HEATING:
      if (temp >= STOP_HEATING && temp < OVERHEAT_TEMP) {
        currentState = STABILIZING;
        digitalWrite(HEATER_PIN, LOW);
      } else if (temp >= OVERHEAT_TEMP) {
        currentState = OVERHEAT;
        digitalWrite(HEATER_PIN, LOW);
      }
      break;

    case STABILIZING:
      if (temp >= STOP_HEATING && temp < OVERHEAT_TEMP) {
        currentState = TARGET_REACHED;
      } else if (temp < START_HEATING) {
        currentState = HEATING;
        digitalWrite(HEATER_PIN, HIGH);
      }
      break;

    case TARGET_REACHED:
      if (temp < START_HEATING) {
        currentState = HEATING;
        digitalWrite(HEATER_PIN, HIGH);
      } else if (temp >= OVERHEAT_TEMP) {
        currentState = OVERHEAT;
        digitalWrite(HEATER_PIN, LOW);
      }
      break;

    case OVERHEAT:
      digitalWrite(HEATER_PIN, LOW);
      break;
  }

  // LED indicator logic (optional)
  digitalWrite(LED_PIN, (currentState == OVERHEAT) ? HIGH : LOW);
}

void logStatus(float temp) {
  Serial.print("Temp: ");
  Serial.print(temp);
  Serial.print("°C | State: ");
  switch (currentState) {
    case IDLE: Serial.println("IDLE"); break;
    case HEATING: Serial.println("HEATING"); break;
    case STABILIZING: Serial.println("STABILIZING"); break;
    case TARGET_REACHED: Serial.println("TARGET_REACHED"); break;
    case OVERHEAT: Serial.println("OVERHEAT"); break;
  }
}
```