**Manual of code**

**Lab 03**

**Explanation of the Code**

This code defines a simple temperature control system using a ModelAgent class. The agent monitors the temperature and controls a heater, turning it on or off based on specific temperature thresholds. It also keeps track of the previous temperature and the current state of the heater to make decisions.

**How the Code Works**

1. The ModelAgent Class

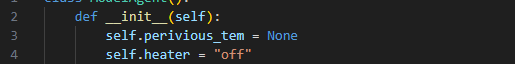
The ModelAgent class is the core of the program. It has three main parts:

* Internal State: The agent remembers the previous temperature and the current state of the heater.
* Methods: The agent has methods to update its state and control the heater.
* Logic: The agent uses simple rules to decide when to turn the heater on or off.

**2. Initial Setup**

When the agent is created, it starts with:

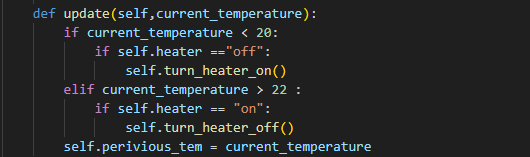
* Previous Temperature: Set to None because no temperature has been recorded yet.
* Heater State: Set to "off" because the heater is turned off by default.

****

**3. Updating the Agent**

The update method is where the agent decides what to do based on the current temperature. Here's how it works:

* If the temperature is below 20°C:
  + The agent checks if the heater is off.
  + If the heater is off, it turns the heater on.
* If the temperature is above 22°C:
  + The agent checks if the heater is on.
  + If the heater is on, it turns the heater off.
* After making a decision, the agent updates the previous temperature to the current temperature.

****

4. Turning the Heater On and Off

The agent has two helper methods to control the heater:

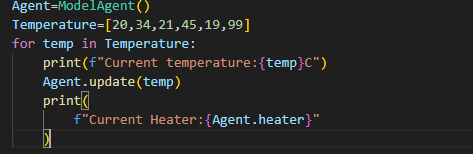
* turn\_heater\_on: Turns the heater on and prints a message.
* turn\_heater\_off: Turns the heater off and prints a message.

****

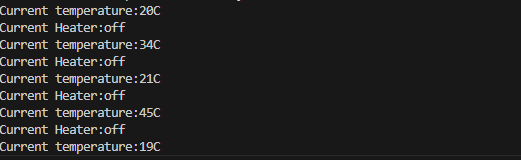
**5. Simulating the Agent**

The code simulates the agent by testing it with a list of temperatures. Here's what happens:

1. The agent is created.
2. A list of temperatures is defined: [20, 34, 21, 45, 19, 99].
3. The agent processes each temperature one by one:
   * It prints the current temperature.
   * It updates the heater's state based on the temperature.
   * It prints the current state of the heater.

****

**Example Output**

****

****