# Exercise 3

What type of agent did you implement in exercise 2?

The agent implemented in Exercise 2 is a reflexive agent. It exhibits simple stimulus-response behavior where each incoming message triggers a predetermined action (increment counter and send response) based on a straightforward condition (counter < 10). It doesn't maintain complex internal state or engage in sophisticated reasoning beyond this basic reaction pattern.

# Exercise 5

Explain the purpose of the separation of Agents and Containers in the mango framework.

The separation of Agents and Containers in the mango framework serves several important purposes:

1. Communication Infrastructure: Containers provide the messaging infrastructure, handling message routing, serialization, and network communication, while agents focus on business logic.
2. Resource Management: Containers manage computational resources and provide a runtime environment for multiple agents, enabling efficient resource sharing.
3. Isolation: Agents are isolated from low-level communication details, allowing developers to focus on agent behavior rather than networking complexities.
4. Scalability: Multiple agents can run within a single container, and multiple containers can be distributed across different machines, supporting horizontal scaling.
5. Lifecycle Management: Containers manage the complete agent lifecycle (creation, registration, activation, termination) in a consistent manner.
6. Interoperability: Containers enable communication between agents running in different environments or on different networks through standardized protocols.

# Exercise 6

The framework mango defines the lifecycle of an Agent. Name and describe every step of this lifecycle and explain the purpose of each. Also, name the types of actions an Agent can start at every phase of its lifecycle.

Agent Lifecycle in mango:

Creation: Agent object is instantiated via constructor

Purpose: Initialize agent with initial state and configuration

Actions: Define instance variables, setup initial conditions, configure agent properties

Registration: Agent is registered with a container using container.register(agent)

Purpose: Make agent addressable and enable message routing

Actions: Container learns about agent's address, agent becomes discoverable by other agents

Activation: Container starts processing messages via mango.activate(container)

Purpose: Begin normal operation where agents can send/receive messages

Actions: Start message processing loops, begin scheduled tasks, enable communication

Message Handling: Agents process incoming messages via handle_message(content, meta)

Purpose: React to communication from other agents

Actions: Process content, update internal state, send responses using schedule_instant_message(), make decisions based on received information

Task Scheduling: Agents can schedule tasks using methods like schedule_instant_message()

Purpose: Enable proactive behavior beyond reactive message handling

Actions: Send messages to other agents, perform periodic actions, execute delayed tasks, initiate communications

Termination: Agent is stopped when container context exits

Purpose: Graceful shutdown and resource cleanup

Actions: Clean up resources, complete pending operations, save state if necessary

# Exercise 8

Given two different problems:

There is a house with two solar panels and one energy storage; the owner solely wants to maximize the self-consumption (consumption of self-produced energy).

There are multiple houses with solar panels, and each wants to sell surplus energy to the energy market and maximize the profit.

Are agents a reasonable approach to these problems? And why (or why not)?

Problem 1 (Single house with solar panels and storage):
Agents are less suitable for this problem. This represents a centralized optimization problem where:

- All components (solar panels, storage, consumption) are physically co-located
- There's a single objective (maximize self-consumption)
- The system components are tightly coupled
- A monolithic controller can efficiently manage all components with global information
- The overhead of agent communication would add unnecessary complexity
- No need for negotiation or distributed decision-making

Problem 2 (Multiple houses selling to energy market):
Agents are highly suitable for this problem because:

- Autonomy: Each house has individual goals (profit maximization)
- Distributed nature: Physically separate entities with local information
- Competition/Cooperation: Houses may compete or form coalitions in energy trading
- Local decision-making: Each agent knows its own generation, consumption, and preferences
- Scalability: Easy to add more houses as autonomous agents

- Adaptability: Agents can respond to changing market conditions individually
- Natural representation: Real-world scenario with multiple independent decision-makers

# Exercise 10

In the lecture, you learned about the PEAS (performance metric, environment, actuators, and sensors) description of agents. Describe every agent of your solution for Exercise 9 following the PEAS categories.

House Agent PEAS Description:

- Performance Metric: Profit maximization (revenue from energy sales minus cost of energy purchases), efficient utilization of self-produced energy
- Environment: Local energy generation/consumption patterns, energy market with fluctuating prices, weather conditions affecting solar production, household energy demands
- Actuators: Submit buy/sell orders to market, adjust offering prices and strategies, control local energy storage (charge/discharge), manage local energy consumption
- Sensors: Receive market price announcements, trade confirmations, monitor own solar generation, track household energy consumption, receive weather forecasts

Market Agent PEAS Description:

- Performance Metric: Market efficiency (successful matches between buyers and sellers), liquidity maintenance, fair price discovery, transaction volume maximization

- Environment: Collection of buy/sell orders from multiple house agents, market regulations and constraints, overall supply-demand balance, price volatility
- Actuators: Match buy/sell orders, broadcast market prices and conditions, send trade confirmations, enforce market rules, clear transactions
- Sensors: Receive energy offers and requests from house agents, monitor market conditions and liquidity, track transaction history, detect market anomalies