import random

class VacuumEnvironment:

"""

Represents a 4-room environment (A, B, C, D) for the vacuum cleaner agent.

0: Clean, 1: Dirty

"""

def \_\_init\_\_(self, initial\_state=None):

# Represent the state as a dictionary: {room: status (0=Clean, 1=Dirty)}

if initial\_state:

self.state = initial\_state

else:

self.state = {'A': random.choice([0, 1]),

'B': random.choice([0, 1]),

'C': random.choice([0, 1]),

'D': random.choice([0, 1])}

self.agent\_location = random.choice(list(self.state.keys())) # Agent starts in a random room

def is\_dirty(self, room):

"""Checks if a room is dirty."""

return self.state.get(room) == 1

def clean(self, room):

"""Cleans a dirty room."""

if room in self.state and self.state[room] == 1:

self.state[room] = 0

return True

return False

def move\_agent(self, new\_room):

"""Moves the agent to a new room."""

if new\_room in self.state:

self.agent\_location = new\_room

return True

return False

def display(self):

"""Prints the current state of the environment."""

print("Environment State:")

for room, status in self.state.items():

agent\_marker = ' V' if room == self.agent\_location else ''

print(f" Room {room}: {'Dirty' if status == 1 else 'Clean'}{agent\_marker}")

print("-" \* 20)

class VacuumAgent:

"""

A simple reflex vacuum cleaner agent for the 4-room environment.

"""

def \_\_init\_\_(self, environment):

self.environment = environment

def sense\_and\_act(self):

"""Senses the environment and performs an action."""

current\_room = self.environment.agent\_location

# Sense

is\_current\_dirty = self.environment.is\_dirty(current\_room)

# Act

if is\_current\_dirty:

print(f"Agent in Room {current\_room}: Cleaning...")

self.environment.clean(current\_room)

else:

print(f"Agent in Room {current\_room}: Moving...")

# Simple random movement to a neighboring room (assuming all rooms are connected)

available\_rooms = list(self.environment.state.keys())

available\_rooms.remove(current\_room) # Cannot move to the same room

if available\_rooms:

next\_room = random.choice(available\_rooms)

self.environment.move\_agent(next\_room)

else:

print("No other rooms to move to.")

# Simulate the agent

# You can optionally set an initial state, e.g., {'A': 1, 'B': 0, 'C': 1, 'D': 0}

env = VacuumEnvironment()

agent = VacuumAgent(env)

print("Initial Environment:")

env.display()

for \_ in range(15): # Run for a few steps

agent.sense\_and\_act()

env.display()

