**Overall Idea**

This is a **simple AI simulation** of a **Reflex Vacuum Cleaner Agent** (a common example from AI textbooks).  
The agent moves between **two rooms** (A and B) and either cleans them or moves to the other room depending on their state.

**Code Walkthrough**

**Import statement**

import random

* **Purpose:**  
  Imports Python's built-in random module to allow:
  + Randomly assigning room states (Clean/Dirty)
  + Randomly deciding if a clean room gets dirty again later
  + Randomly choosing where the agent starts.

**Environment Class**

class Environment:

* Represents the **world** the agent lives in.
* **Stores:**
  + Room states (Clean or Dirty)
  + Agent's current location (A or B)

**Constructor (Initialization)**

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

self.rooms = {'A': random.choice(['Clean', 'Dirty']),

'B': random.choice(['Clean', 'Dirty'])}

self.agent\_location = random.choice(['A', 'B'])

* Runs **once** when you create an Environment object.
* self.rooms: Dictionary storing the state of both rooms.
  + Each room (A and B) is randomly set to "Clean" or "Dirty".
* self.agent\_location: Randomly set to 'A' or 'B'.

**Check if Room is Dirty**

def is\_dirty(self, location):

return self.rooms[location] == 'Dirty'

* **Input:** Room location ('A' or 'B').
* **Output:** True if that room is "Dirty", otherwise False.
* **Purpose:** Lets the agent know whether it should clean or move.

**Clean a Room**

def clean(self, location):

self.rooms[location] = 'Clean'

* Changes the state of a given room to "Clean".

**Move Agent**

def move(self, new\_location):

self.agent\_location = new\_location

* Updates the agent’s location to 'A' or 'B'.

**Randomly Dirty Rooms Again**

def randomly\_dirty\_rooms(self, probability=0.3):

for room in self.rooms:

if self.rooms[room] == 'Clean' and random.random() < probability:

self.rooms[room] = 'Dirty'

print(f"Environment: Room {room} got dirty again!")

* **Purpose:** Simulates dirt reappearing over time.
* Goes through each room:
  + If it’s currently "Clean" AND a random number is less than probability (default 0.3 = 30% chance),  
    → it becomes "Dirty" again.
* Also prints a message when dirt appears.

**Display Current Environment**

def display(self):

print(f"Agent is in room {self.agent\_location}")

print(f"Room A: {self.rooms['A']}, Room B: {self.rooms['B']}")

* Shows:
  + Agent’s current location
  + The state of each room

**ReflexVacuumAgent Class**

class ReflexVacuumAgent:

* Represents the **AI agent**.
* Follows a **simple reflex rule**:
  + If current room is dirty → clean it
  + Else → move to the other room

**Decision and Action**

def act(self, env: Environment):

location = env.agent\_location

if env.is\_dirty(location):

print(f"Action: Suck dirt in Room {location}")

env.clean(location)

elif location == 'A':

print("Action: Move Right to Room B")

env.move('B')

else:

print("Action: Move Left to Room A")

env.move('A')

* **Input:** The Environment object.
* Steps:
  1. Get the agent’s current location.
  2. If current room is dirty → clean it.
  3. Else:
     + If in Room A → move to Room B.
     + If in Room B → move to Room A.

**Main Program Execution**

env = Environment()

agent = ReflexVacuumAgent()

* Creates:
  + An Environment object with random starting conditions.
  + A ReflexVacuumAgent object.

**Run Simulation for 10 Steps**

for step in range(10):

print(f"\nStep {step + 1}")

env.display()

agent.act(env)

env.randomly\_dirty\_rooms(probability=0.3)

* Loop runs **10 times** (10 time steps):
  1. Print step number.
  2. Show environment state.
  3. Let the agent act (clean or move).
  4. Randomly dirty rooms again.

**What is this called?**

* **In AI terms:** This is a **Simple Reflex Agent** in a **Vacuum Cleaner World** simulation.
* **Reason:** The agent decides what to do **only based on the current percept** (current location + dirt status), without memory or planning.

If you want, I can **draw a flowchart** showing how the agent decides at each step.  
That would make the process visually clear.