QUESTION 1 [26 MARKS]

Your task is to simulate the simple robot vacuum agent that use at home to clean the the house. The purposed of the agent is to make sure the battery life of the robot vacuum is enough to complete it's task. The agent will give an order to the vacuum to return to the charging station if the battery is less than 30% and stop the charging when the battery is 100%. Table 1 shows the table of robot vacuum battery stage.

Table 1: Robot vacuum percentage and stage.

Battery indicator (%)	Battery stage
0 – 30	Low
31 – 80	Moderate
81 – 100	High

a) Suggest what the best **Agent Program** to accomplish the task. Explain your answer.
[4 Marks]

The best agent program for the robot vacuum is **a simple reflex agent** with condition-action rules. This is because the robot vacuum's task is simple:
First it must respond to the environment by checking the battery level and then either continue cleaning or return to the charging station. The agent will use sensors to observe the battery level and make decisions based on predefined rules (e.g., return to the charging dock when battery is below 30%, stop charging when the battery is at 100%). Reflex agents are effective for such tasks because they follow direct mappings of conditions to actions without the need for complex planning or learning.

b) Suggest **TWO (2)** improvement/features to the agent, so that it can be a better agent program compared to the agent in **Question 2 (a)**.
[5 Marks]

Two improvements/features to the agent, so that it can be a good boy agent:

1) **Path Optimization for Energy Efficiency**
   The robot vacuum can be improved by adding a path-planning feature to optimize its movements. Instead of randomly cleaning the house, it can map the environment and choose the most efficient path to cover the area in the least amount of time and with minimal battery consumption.
2) **Adaptive Cleaning Mode**
   The agent can have an adaptive mode where it adjusts its cleaning speed and suction power based on the dirt levels so it can save on battery from adjusting the environment to vacuum. For instance, if it detects a high level of dirt, it can increase

its suction power and clean more thoroughly, but if the dirt level is low, it can conserve energy by lowering its suction power and moving faster.

c) Write a Pseudo Code or Code Snipped (in any programming language) for condition action rule for the robot vacuum problem above.

[3 Marks]

Some condition action rules are based on perception. Namely clean or dirty and where the dirt is. The robot vacuum will suck up dust with the help of energy from the battery.

battery_sensor.py

```python
class BatterySensor:
    """
    A sensor class to monitor the battery level of the robot vacuum.
    """
    def __init__(self):
        self.battery_percentage = 100

    def drain(self, usage_rate):
        """
        Simulate the battery drain during cleaning tasks.
        The battery decreases by usage_rate% every call.
        """
        if self.battery_percentage > 0:
            self.battery_percentage = max(0, self.battery_percentage - usage_rate)

    def charge(self, charge_rate=5):
        """
        Simulate the charging process.
        The battery increases by charge_rate% every call.
        """
        if self.battery_percentage < 100:
            self.battery_percentage = min(100, self.battery_percentage + charge_rate)

    def get_battery_status(self):
        """
        Returns the battery level in percentage.
        """
        return self.battery_percentage

    def get_battery_stage(self):
        """
        Returns the stage of the battery based on the current percentage.
        """
        if 0 <= self.battery_percentage <= 30:
            return "Low"
        elif 31 <= self.battery_percentage <= 80:
            return "Moderate"
        else:
            return "High"
```

location_sensor.py

```python
class LocationSensor:
    """
    A sensor class to track the robot's location (cleaning area or charging station).
    """
    def __init__(self):
        self.location = "Charging Station"

    def move_to_cleaning_area(self):
        self.location = "Cleaning Area"

    def move_to_charging_station(self):
        self.location = "Charging Station"

    def get_location(self):
        return self.location

```

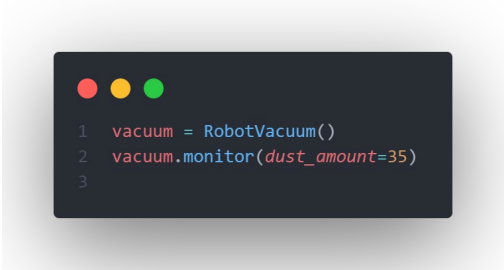
dust_sensor.py

```python
class DustSensor:
    """
    A sensor class to monitor the amount of dust in the cleaning area.
    It returns a dust level that will adjust the suction power of the vacuum.
    """
    def __init__(self):
        self.dust_level = 0

    def detect_dust(self, dust_amount):
        """
        Simulate dust detection.
        Higher values indicate more dust.
        """
        self.dust_level = dust_amount

    def get_dust_level(self):
        """
        Returns the current dust level.
        """
        return self.dust_level

```

robot_vacuum.py

```python
from battery_sensor import BatterySensor
from location_sensor import LocationSensor
from dust_sensor import DustSensor

class RobotVacuum:
    """
    A robot vacuum agent that manages its battery and cleaning tasks efficiently.
    The vacuum adjusts its suction speed based on dust sensor readings.
    """
    def __init__(self):
        self.battery_sensor = BatterySensor()
        self.location_sensor = LocationSensor()
        self.dust_sensor = DustSensor()
        self.cleaning_task = False
        self.suction_speed = 1

    def adjust_suction_speed(self):
        """
        Adjust the suction speed based on the dust level detected by the dust sensor.
        Higher dust levels increase suction speed.
        """
        dust_level = self.dust_sensor.get_dust_level()
        if dust_level <= 10:
            self.suction_speed = 1
        elif 10 < dust_level <= 50:
            self.suction_speed = 2
        else:
            self.suction_speed = 3
        print(f"Adjusted suction speed to {self.suction_speed} due to dust level {dust_level}")

    def start_cleaning(self, dust_amount):
        """
        Starts the cleaning task if the battery is sufficient.
        """
        if self.battery_sensor.get_battery_status() > 30:
            self.cleaning_task = True
            self.location_sensor.move_to_cleaning_area()
            self.dust_sensor.detect_dust(dust_amount)
            self.adjust_suction_speed()
            print(f"Vacuum started cleaning. Battery at {self.battery_sensor.get_battery_status()}%")
        else:
            print(f"Can't start cleaning. Battery too low: {self.battery_sensor.get_battery_status()}%")

    def stop_cleaning(self):
        """
        Stops the cleaning task.
        """
        self.cleaning_task = False
        print("Vacuum stopped cleaning.")

    def return_to_charging_station(self):
        """
        Orders the robot to return to the charging station.
        """
        self.location_sensor.move_to_charging_station()
        print(f"Returning to the charging station. Battery at {self.battery_sensor.get_battery_status()}%")

    def charge_battery(self):
        """
        Charges the battery until it reaches 100%.
        """
        if self.location_sensor.get_location() == "Charging Station":
            while self.battery_sensor.get_battery_status() < 100:
                self.battery_sensor.charge()
                print(f"Charging... Battery at {self.battery_sensor.get_battery_status()}%")
            print("Battery fully charged.")
        else:
            print("Cannot charge. The vacuum is not at the charging station.")

    def monitor(self, dust_amount):
        """
        Continuously monitors the battery and manages tasks.
        """
        for _ in range(50):
            battery_status = self.battery_sensor.get_battery_status()
            battery_stage = self.battery_sensor.get_battery_stage()
            location = self.location_sensor.get_location()

            print(f"Battery Stage: {battery_stage}, Location: {location}")

            if self.cleaning_task:
                if battery_status <= 30:
                    print("Battery low. Stopping cleaning and returning to the charging station.")
                    self.stop_cleaning()
                    self.return_to_charging_station()
                    self.charge_battery()
                else:
                    drain_rate = self.suction_speed * 2
                    self.battery_sensor.drain(drain_rate)
            else:
                if location == "Charging Station" and battery_status < 100:
                    self.charge_battery()
                elif battery_status > 30:
                    self.start_cleaning(dust_amount)

```

Example usage:



d) Explain the nature of the robot vacuum agent environment as listed below.

i) Fully or Partially Observable

The environment is **partially observable**. The robot vacuum can only observe certain aspects of the environment, like battery life and immediate obstacles, but it may not have complete information about the entire house layout or the location of all dirt in advance.

ii) Single or Multi Agent

It is a **single-agent** system. The robot vacuum operates independently without interacting with other agents or robots.

iii) Deterministic or Stochastic

The environment is **stochastic** because actions like moving across a room or avoiding obstacles may have uncertain outcomes. The robot might encounter unexpected dirt levels or objects that it must navigate around, adding variability to its task.

iv) Episodic or Sequential

The task is **sequential** because the actions the robot takes (cleaning, recharging, etc.) affect future states. Decisions made now (like running low on battery) will influence the need to recharge later.

v) Static or Dynamic

The environment is **dynamic** because it can change while the robot is operating. For example, people might move objects or create new obstacles in the robot's cleaning path while it is working.

vi) Discrete or Continuous

The environment is **continuous** because the robot operates in a physical space with continuous movement and measurements. Its battery levels and the physical location of obstacles are not limited to distinct, discrete values.

vii) Known or Unknown

The environment is **unknown** at the start. The robot may not have complete information about the house layout or obstacles initially and must use sensors to navigate and adapt to the surroundings.

[14 Marks]

QUESTION 2 [24 MARKS]

a) What is the difference between Strong and Weak AI, and how does their level of intelligence and capabilities differ?

[6 Marks]

The difference between Strong AI and Weak AI lies primarily in the depth of their capabilities, particularly in reasoning, problem-solving, and self-awareness:

Strong AI	Weak AI
• his perspective asserts that it is possible to build machines with real intelligence that mirrors human cognitive functions. Strong AI systems would not only be capable of performing tasks but also possess consciousness, self-awareness, and the ability to reason independently. • They could think about problems and come up with solutions autonomously, develop their own values and worldviews, and have survival instincts like living beings. Such machines would operate at the level of creating new civilizations, essentially replicating human-like cognition and behaviour in machines. • However, no such systems exist today, as we have not yet achieved the scientific breakthroughs required to create machines with this level of intelligence.	• In contrast, weak AI refers to systems that appear intelligent but lack true understanding or consciousness. • These machines can simulate human behaviour in specific tasks but are not genuinely capable of reasoning or self-awareness. • They can process data and solve problems but do so through predefined algorithms and rules rather than independent thought. For example, programs like Watson or AlphaGo demonstrate intelligent behaviour in games or specialized applications but do not possess true cognition.

Their level of intelligence, strong AI possesses human-like reasoning and consciousness, whereas weak AI is limited to performing specific tasks without genuine understanding.
Capabilities difference, Strong AI could independently generate values, worldviews, and instincts for survival, whereas weak AI merely simulates intelligent behaviour without true reasoning or awareness.

b) Provide an explanation of whether ChatGPT falls under the category of strong or weak artificial intelligence?

[3 Marks]

ChatGPT falls under the category of Weak AI. While it can generate human-like text and assist with various tasks, its capabilities are limited to multimodal like text-based, images-based, or voice-based interactions. It does not have true understanding or consciousness, nor can it perform tasks beyond the scope of language generation and processing**.**

But now, ChatGPT has started to move towards AGI from their mission, that is it now has reasoning (o1-mini and o1-preview), where the AI can simulate thinking first before giving answers to the user.

ChatGPT already has a real time feature (Advanced Voice Mode), where AI can simulate a human-like voice, multi-lingual and be answered quickly like a human conversation. In this mode, can express such as sad, happy, surprised or other expressions.

c) What do you understand about Artificial Intelligence?

[3 Marks]

Artificial Intelligence (AI) refers to the development of computer systems that can perform tasks requiring human intelligence, such as recognizing speech, understanding natural language, making decisions, and learning from data. AI can be rule-based, where it follows predefined rules, or it can involve machine learning, where it improves its performance through experience and data.

d) In your opinion, what do you foresee Artificial Intelligence will be in the future. Your answer must be based on the current trend and issues in Artificial Intelligence.

[4 Marks]

In the future, I foresee that Artificial Intelligence (AI) will play an even more integrated and transformative role in entertainment and digital media, particularly within niche communities such as the anime and Vtuber spaces. Currently, we are witnessing the rise of AI-driven virtual personalities like Neuro-Sama (Neuro-sama - YouTube), which serve as early indicators of how AI might evolve to replace or enhance Vtubers.

As AI continues to develop, I believe it will not only simulate human-like interactions but also possess a form of advanced understanding or "consciousness" that mimics human behaviours and emotions. For example, AI Vtubers could be capable of interacting with fans in more complex and emotionally nuanced ways, adapting to their preferences and creating personalized experiences. This could result in AI-generated anime characters that blur the lines between fiction and reality, with characters potentially appearing to have self-awareness.

ONE OF THE KEY REASON for my **strong interest in AI** stems from the valuable insights, so I'm joining my study of Artificial Intelligence Techniques. I aspire to leverage AI technology to enhance the realism and immersion of anime, pushing the boundaries of digital creativity. Additionally, I have begun exploring Vixevia

(https://github.com/IRedDragonICY/vixevia) to gradually bring anime to life through AI-driven innovations.

e) Offsetting your carbon footprint (excessive carbon emissions) while travelling can be difficult to manage, but this is achievable with the advancement of AI technology. How can this problem be achieved? Discuss your idea.

[4 Marks]

AI can help offset carbon footprints in travel by optimizing flight routes and schedules to minimize fuel consumption. It can also assist in predicting the most environmentally friendly transport modes and suggest low-emission alternatives, such as trains over planes for short distances. AI-based carbon offsetting tools could calculate the traveller's carbon footprint and recommend effective offset methods, such as supporting reforestation or renewable energy projects.

f) Give one real world example of **Turing Test application**. Explain your answer

[4 Marks]

A real-world example of the Turing Test's application is the Loebner Prize, an annual competition that awards prizes for chatbots that can best mimic human conversation. The competition is designed to apply the principles of the Turing Test by having human judges converse with both humans and chatbots through text interfaces. The goal for the chatbot is to deceive the judges into thinking it is human.

In this scenario, like the Turing Test's phase 2, the computer programs (chatbots) are tasked with generating responses that reflect human-like reasoning, emotional understanding, and even deliberate mistakes to simulate imperfections in human communication. The judges ask questions on a variety of topics, ranging from everyday life to complex emotional scenarios, to evaluate how convincingly the chatbot can replicate human behaviour. If a chatbot manages to fool enough judges into believing it is human, it is considered successful in passing this Turing-style test.

This practical application demonstrates the use of AI systems in conversational tasks and reflects the principles laid out in Turing's original concept: the ability of a machine to exhibit intelligent behaviour indistinguishable from a human in a text-based setting.