

Task 3

1. The Class Specifications

In UML, we use specific notation for visibility: - for **private** attributes and + for **public** methods.

Astronaut

Attributes	Methods
- id: int	+ assignMission(mission: SpaceMission): void
- name: String	
- specialization: String	
- experienceYears: int	

SpaceMission

Attributes	Methods
- id: int	+ addCrewMember(astronaut: Astronaut): void
- missionName: String	+ removeCrewMember(astronaut: Astronaut): void
- destination: String	+ isReady(): boolean
- duration: int	

- crew: List<Astronaut>	
----------------------------	--

Gemini said

Designing a UML Class Diagram involves organizing the classes, their internal data, and how they relate to one another. Since a **SpaceAgency** manages both missions and astronauts, and **SpaceMissions** are composed of astronauts, your diagram will show a clear hierarchy.

1. The Class Specifications

In UML, we use specific notation for visibility: - for **private** attributes and + for **public** methods.

Astronaut

Attributes	Methods
- id: int	+ assignMission(mission: SpaceMission): void
- name: String	
- specialization: String	
- experienceYears: int	

SpaceMission

Attributes	Methods
- id: int	+ addCrewMember(astronaut: Astronaut): void
- missionName: String	+ removeCrewMember(astronaut: Astronaut): void
- destination: String	+ isReady(): boolean
- duration: int	

```
- crew: List<Astronaut>
```

SpaceAgency

Attributes	Methods
- agencyName: String	+ launchMission(mission: SpaceMission): void
- budget: double	+ recruitAstronaut(astronaut: Astronaut): void
- missions: List<SpaceMission>	+ terminateMission(mission: SpaceMission): void
- astronauts: List<Astronaut>	

2 To get full marks on a UML assignment, you need to show the connections (associations) correctly:

1. **Aggregation (Open Diamond):** Draw a line with an open diamond from SpaceAgency to SpaceMission and Astronaut. This shows that the Agency *contains* them, but they can exist independently.
2. **Association (Simple Line):** Draw a line between SpaceMission and Astronaut.
3. **Multiplicity:**
 - a. One **SpaceAgency** to many (\$1 \dots *) **SpaceMissions**.
 - b. One **SpaceMission** to many (\$1 \dots *) **Astronauts** (representing the crew).

3 1. The Structure of the Robot Class Diagram

In UML notation, we use specific symbols to denote **Visibility**:

- - (Minus) represents private fields.
- + (Plus) represents public methods.

Section	UML Notation
Class Name	Robot
Attributes	- model: String - batteryLife: int - purpose: String
Methods	+ Robot(model: String, batteryLife: int, purpose: String) + performTask(): void + recharge(): void + isFullyCharged(): boolean