

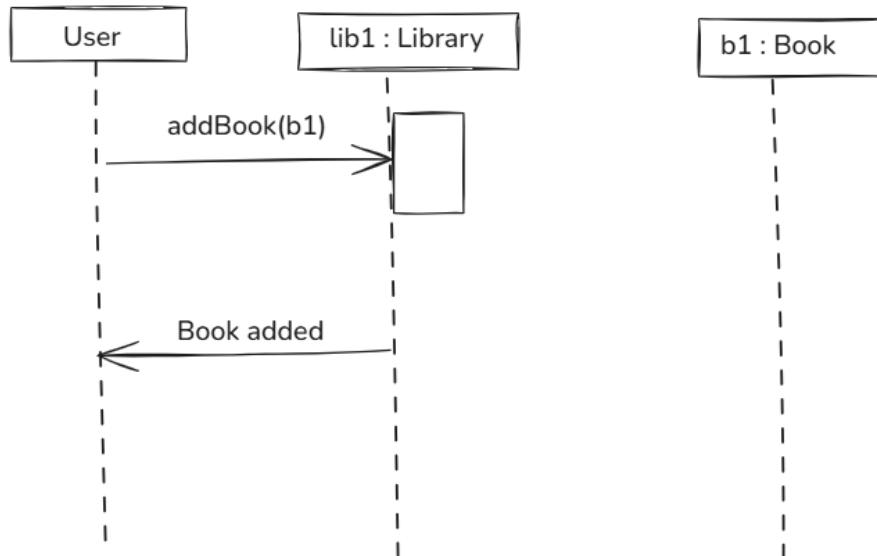
## Assisted Problems

### Problem 1: Library and Books (Aggregation)

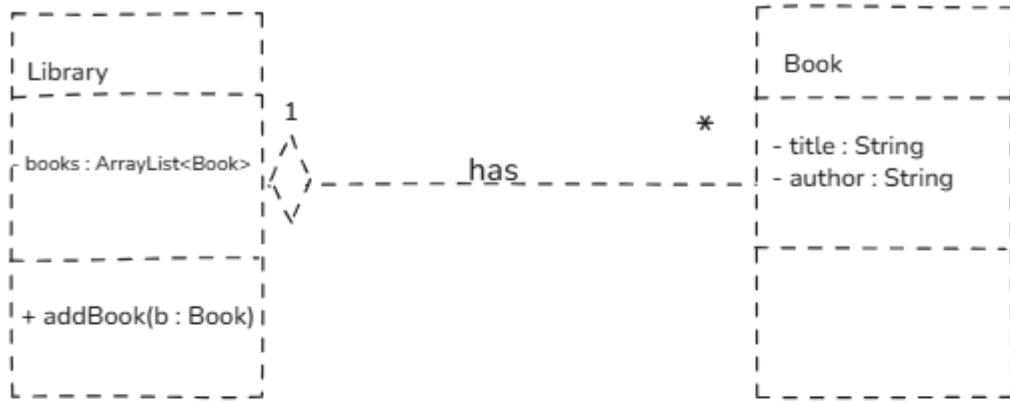
- **Description:** Create a Library class that contains multiple Book objects. Model the relationship such that a library can have many books, but a book can exist independently (outside of a specific library).
- **Tasks:**
  - Define a Library class with an ArrayList of Book objects.
  - Define a Book class with attributes such as title and author.
  - Demonstrate the aggregation relationship by creating books and adding them to different libraries.
- **Goal:** Understand aggregation by modeling a real-world relationship where the Library

#### Sequence Diagram

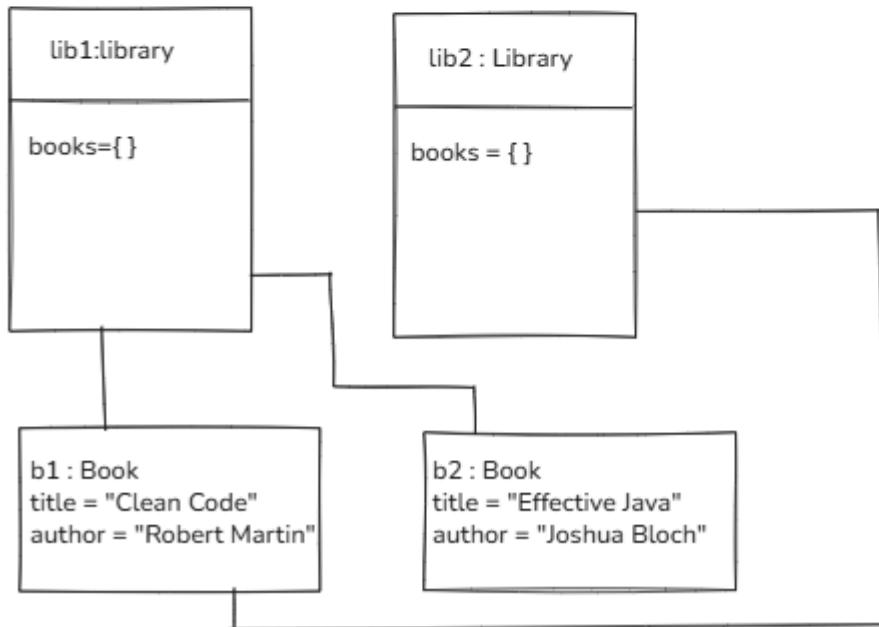
Sequence Diagram – Add Book to Library



## class Diagram



## Object diagram



## Self Problems

### Problem 1: School and Students with Courses (Association and Aggregation)

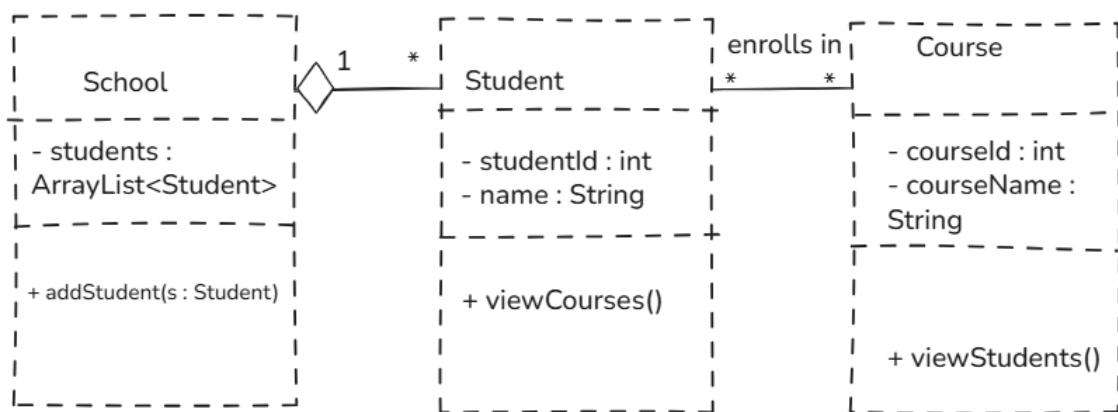
- **Description:** Model a School with multiple Student objects, where each student can enroll in multiple courses, and each course can have multiple students.
- **Tasks:**
  - Define School, Student, and Course classes.
  - Model an association between Student and Course to show that students can enroll in multiple courses.
  - Model an aggregation relationship between School and Student.
  - Demonstrate how a student can view the courses they are enrolled in and how a course can show its enrolled students.

**Goal:** Practice association by modeling many-to-many relationships between students and courses.

## Self Problems

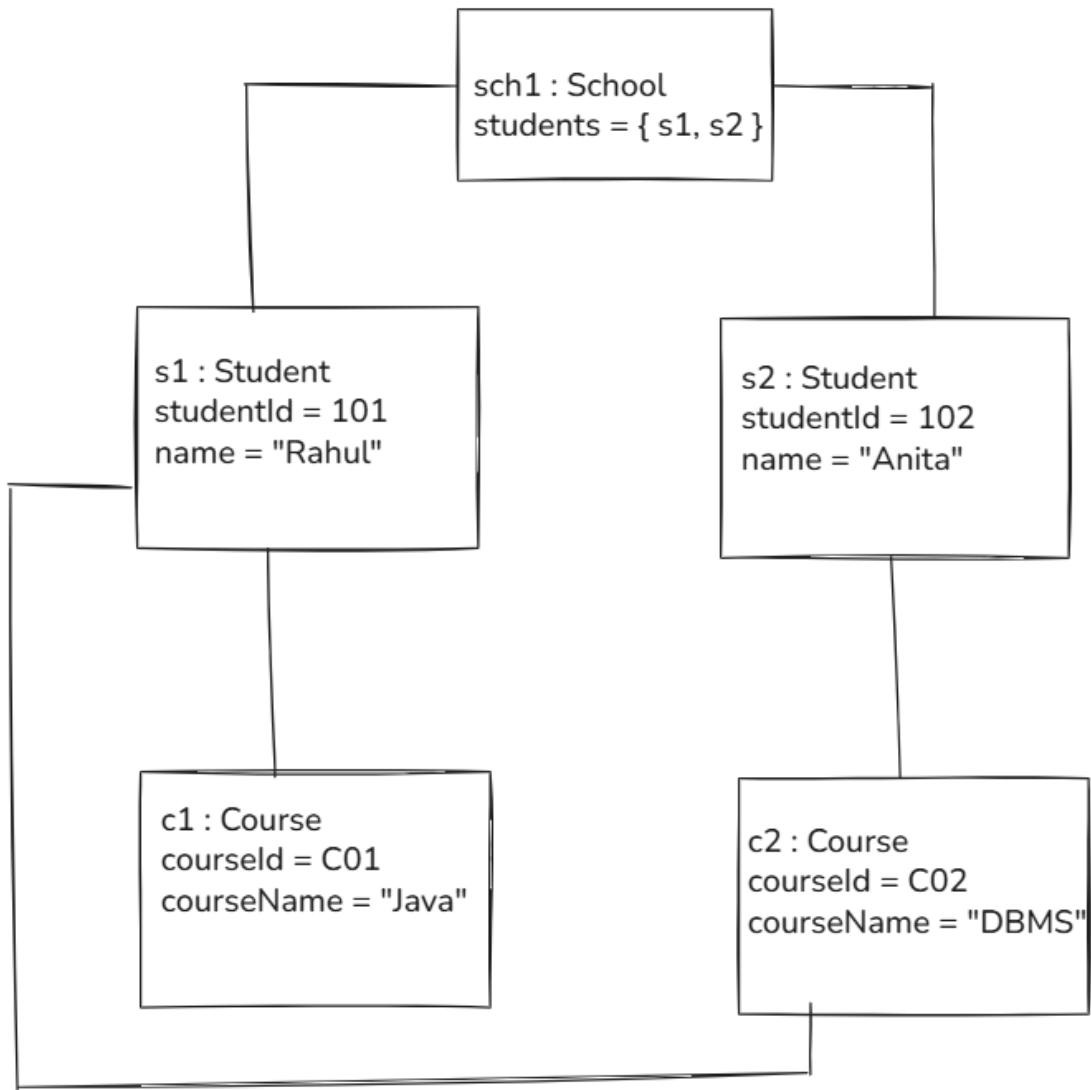
Problems 1

Class Diagram

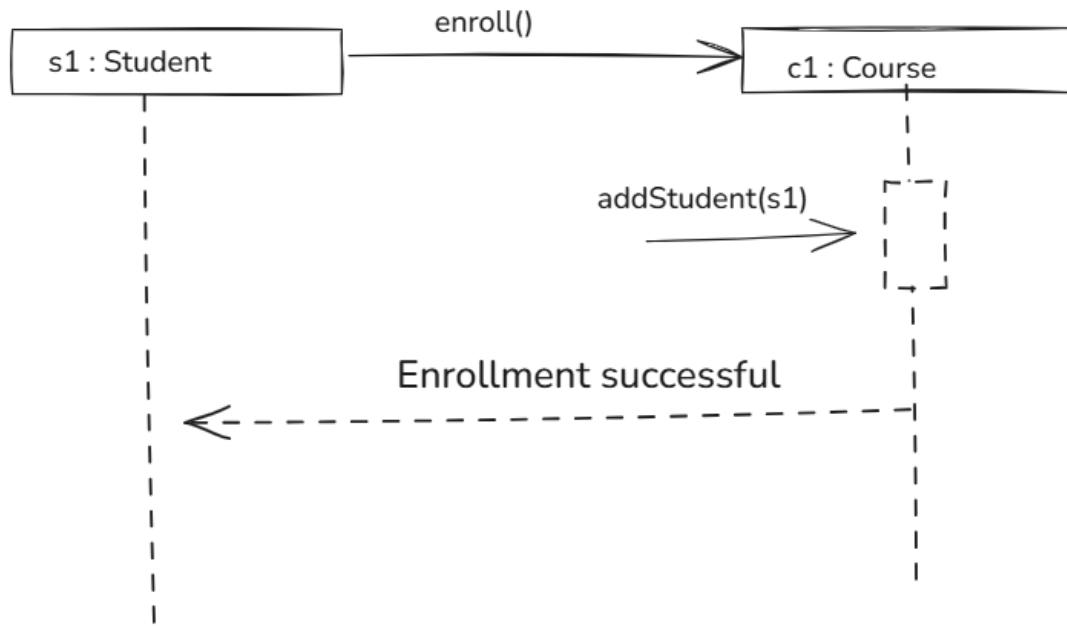


Move [Ctrl] and double-click or press [Ctrl + Enter] to edit points

## Object Diagram



### Sequence Diagram – Student Enrolls in Course

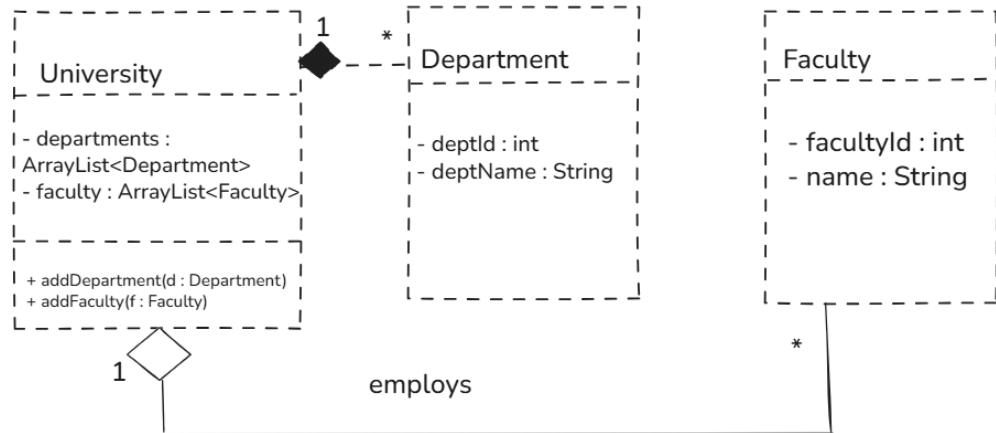


### Problem 2: University with Faculties and Departments (Composition and Aggregation)

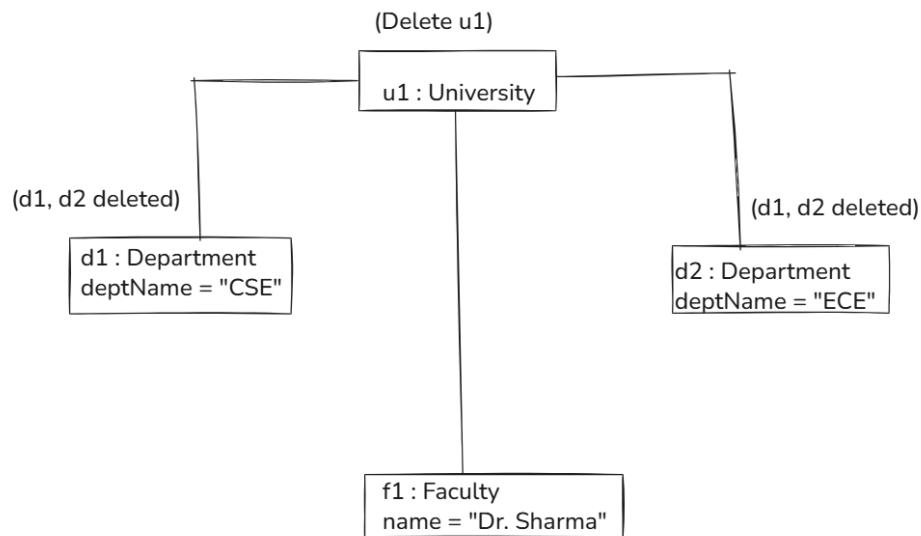
- **Description:** Create a University with multiple Faculty members and Department objects. Model it so that the University and its Departments are in a composition relationship (deleting a university deletes all departments), and the Faculty members are in an aggregation relationship (faculty can exist outside of any specific department).
- **Tasks:**
  - Define a University class with Department and Faculty classes.
  - Demonstrate how deleting a University also deletes its Departments.
  - Show that Faculty members can exist independently of a Department.
- **Goal:** Understand the differences between composition and aggregation in modeling complex hierarchical relationships.

## Problem 2: university with Faculties and Departments

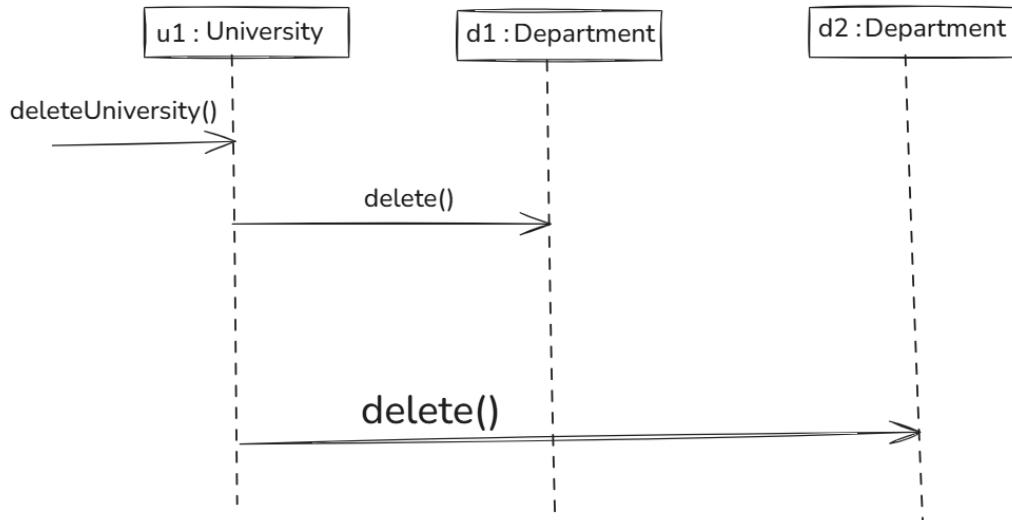
Class diagram .



Object diagram



Sequence Diagram

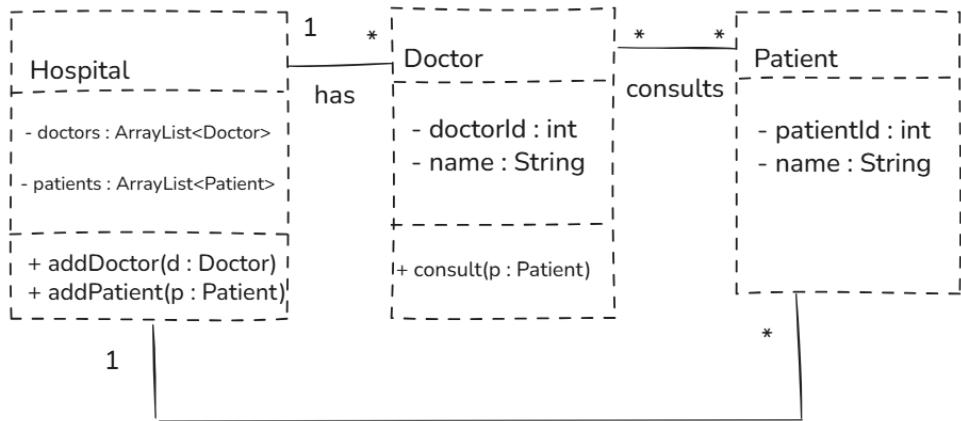


### Problem 3: Hospital, Doctors, and Patients (Association and Communication)

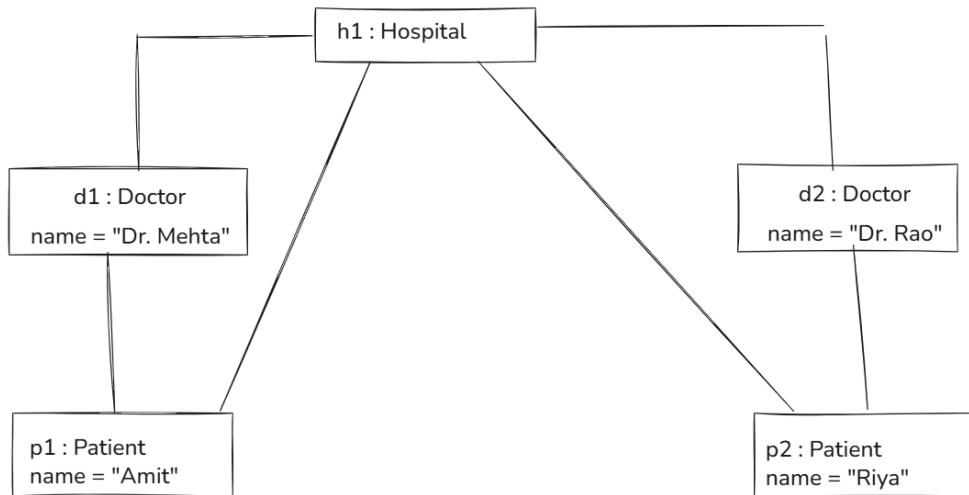
- **Description:** Model a Hospital where Doctor and Patient objects interact through consultations. A doctor can see multiple patients, and each patient can consult multiple doctors.
- **Tasks:**
  - Define a Hospital class containing Doctor and Patient classes.
  - Create a method `consult()` in the Doctor class to show communication, which would display the consultation between a doctor and a patient.
  - Model an association between doctors and patients to show that doctors and patients can have multiple relationships.
- **Goal:** Practice creating an association with communication between objects by modeling doctor-patient consultations.

### Problem 3: Hospital, Doctors, and Patients (Association and Communication)

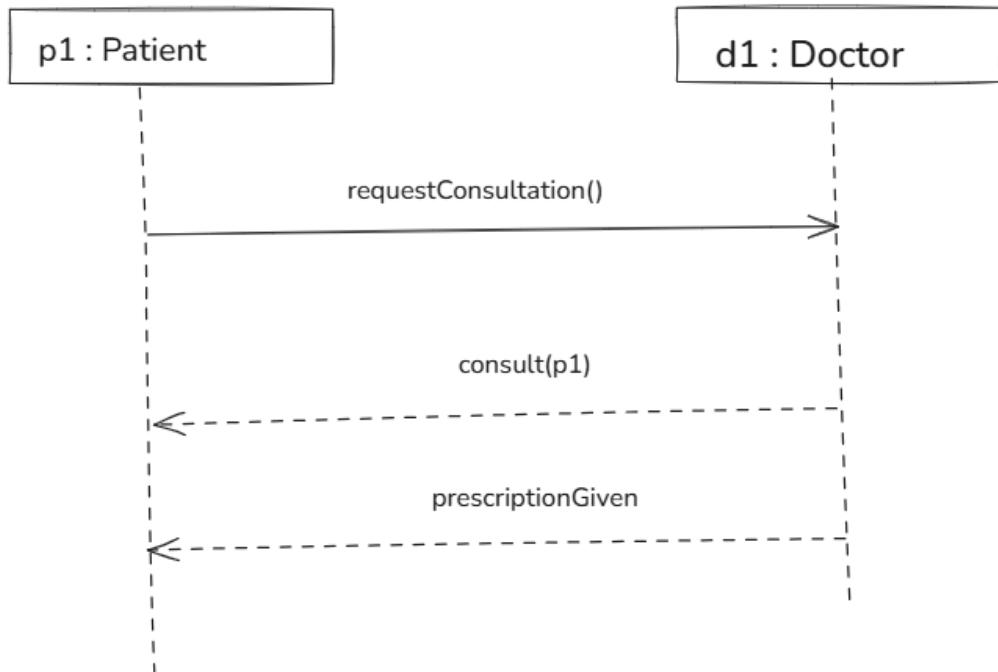
Class diagram .



Object Diagram



### Sequence Diagram

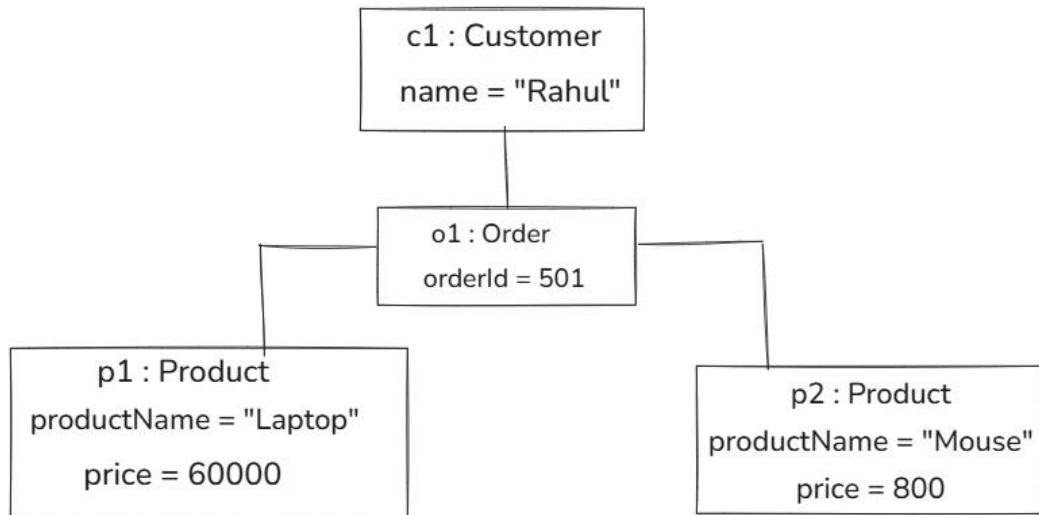


### Problem 4: E-commerce Platform with Orders, Customers, and Products

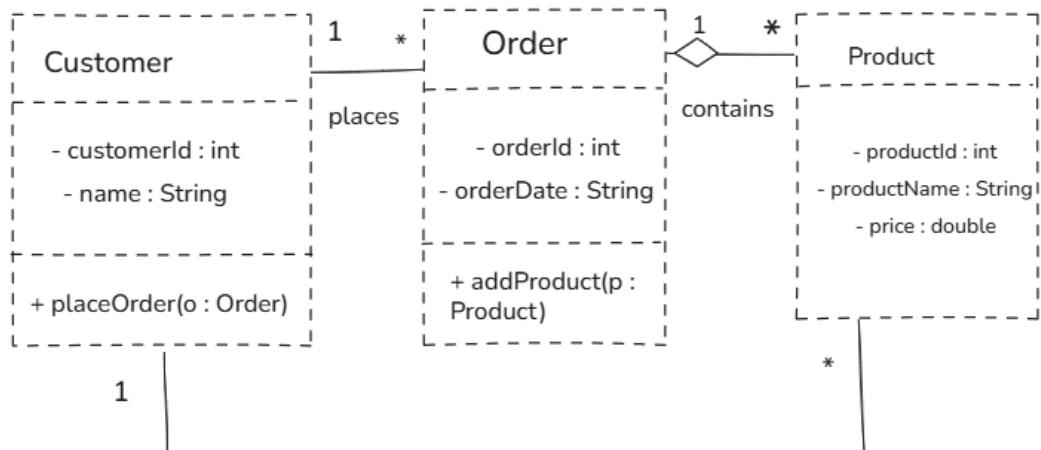
- **Description:** Design an e-commerce platform with Order, Customer, and Product classes. Model relationships where a Customer places an Order, and each Order contains multiple Product objects.

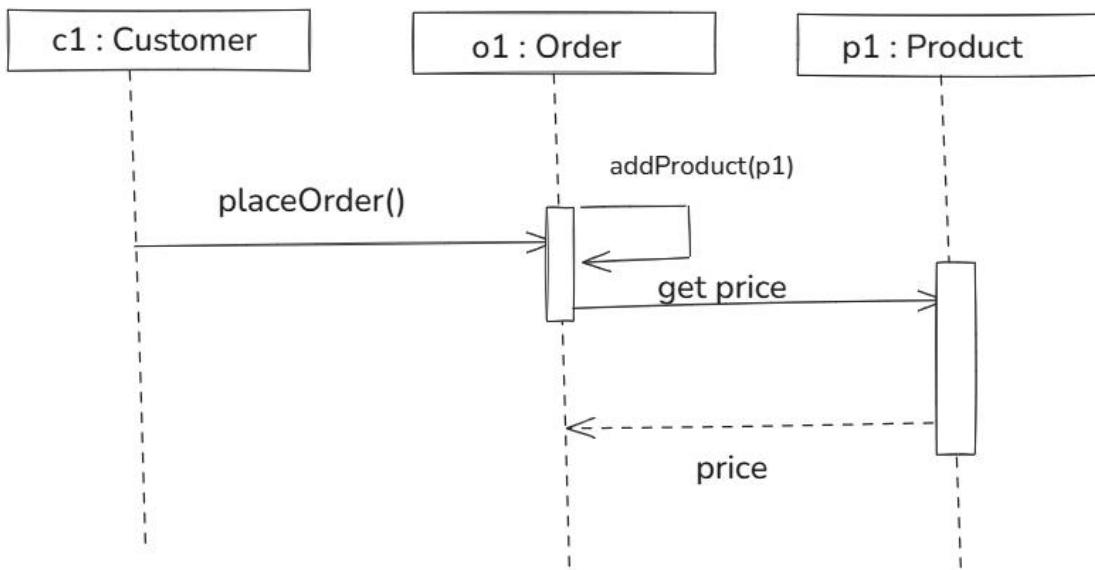
**Goal:** Show communication and object relationships by designing a system where customers communicate through orders, and orders aggregate products.

Object diagram



Class diagram .



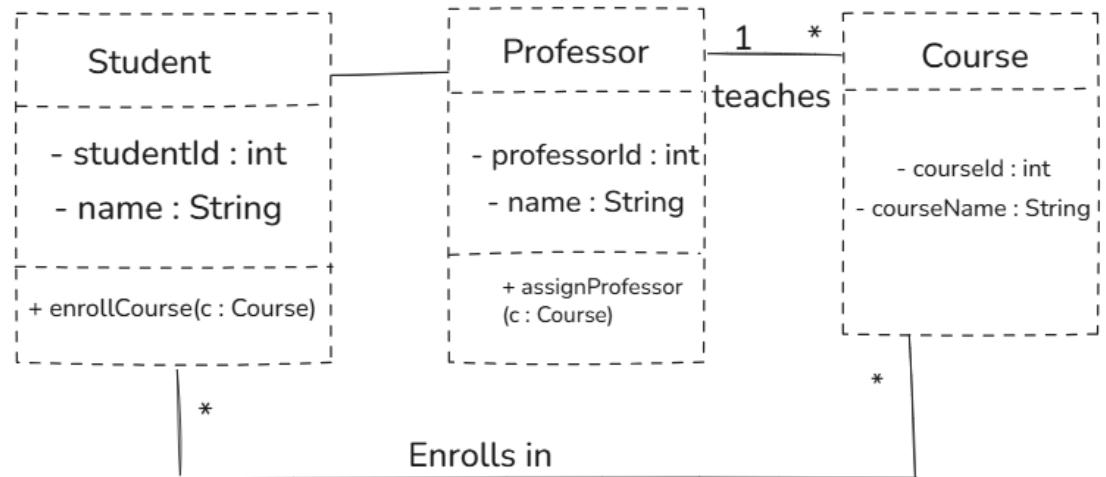


### Problem 5: University Management System

- **Description:** Model a university system with Student, Professor, and Course classes. Students enroll in courses, and professors teach courses. Ensure students and professors can communicate through methods like `enrollCourse()` and `assignProfessor()`.
- **Goal:** Use association and aggregation to create a university system that emphasizes relationships and interactions among students, professors, and courses.

## problem 5

Class diagram .



Object diagram

