Aggregation

🎯 **Direct Challenges**

1. Create Department and Student classes showing aggregation.

```
class Department {
    String deptName;

    Department(String deptName) {
        this.deptName = deptName;
    }
}

class Student {
    String name;
    Department department;

    Student(String name, Department department) {
        this.name = name;
        this.department = department;
    }

    void display() {
        System.out.println(name + " belongs to " + department.deptName + " department.");
    }
}
```

```
public class Main {
    public static void main(String[] args) {
        Department d = new Department("Computer Science");
        Student s = new Student("Vani", d);
        s.display();
    }
}
```

Output:

Vani belongs to Computer Science department.

2. Model a Team that contains players as aggregated objects.

```
class Player {
    String name;

    Player(String name) {
        this.name = name;
    }
}

class Team {
    String teamName;
    Player[] players;

    Team(String teamName, Player[] players) {
        this.teamName = teamName;
        this.players = players;
    }
```

```
    void displayTeam() {
        System.out.println("Team: " + teamName);
        for (Player p : players) {
            System.out.println("- " + p.name);
        }
    }
}

public class Main {
    public static void main(String[] args) {
        Player[] players = { new Player("Asha"), new Player("Bhavya"), new Player("Chitra") };
        Team t = new Team("Warriors", players);
        t.displayTeam();
    }
}
```

Output:

```
Team: Warriors
- Asha
- Bhavya
- Chitra
```

3. Illustrate aggregation with Teacher and Subject.

```
class Subject {
    String subjectName;

    Subject(String subjectName) {
        this.subjectName = subjectName;
    }
```

```
}

class Teacher {
    String name;
    Subject subject;

    Teacher(String name, Subject subject) {
        this.name = name;
        this.subject = subject;
    }

    void display() {
        System.out.println(name + " teaches " + subject.subjectName);
    }
}

public class Main {
    public static void main(String[] args) {
        Subject s = new Subject("Mathematics");
        Teacher t = new Teacher("Mr. Kumar", s);
        t.display();
    }
}
```

Output:

Mr. Kumar teaches Mathematics

🧩 **Scenario-Based Challenges**

1. Build a University class that aggregates multiple College objects.

```
class College {
    String name;

    College(String name) {
        this.name = name;
    }
}

class University {
    String uniName;
    College[] colleges;

    University(String uniName, College[] colleges) {
        this.uniName = uniName;
        this.colleges = colleges;
    }

    void showColleges() {
        System.out.println("University: " + uniName);
        for (College c : colleges) {
            System.out.println("- " + c.name);
        }
    }
}

public class Main {
    public static void main(String[] args) {
```

```
        College[] list = {
            new College("Engineering College"),
            new College("Arts College"),
            new College("Medical College")
        };
        University u = new University("Andhra University", list);
        u.showColleges();
    }
}
```

Output:

```
University: Andhra University
- Engineering College
- Arts College
- Medical College
```

2. Create a Hospital with a list of Doctor objects (not tightly bound).

```
class Doctor {
    String doctorName;

    Doctor(String doctorName) {
        this.doctorName = doctorName;
    }
}

class Hospital {
    String name;
    List<Doctor> doctors;
```

```
    Hospital(String name, List<Doctor> doctors) {
        this.name = name;
        this.doctors = doctors;
    }

    void showDoctors() {
        System.out.println("Hospital: " + name);
        for (Doctor d : doctors) {
            System.out.println("- Dr. " + d.doctorName);
        }
    }
}

import java.util.*;

public class Main {
    public static void main(String[] args) {
        List<Doctor> docList = new ArrayList<>();
        docList.add(new Doctor("Anjali"));
        docList.add(new Doctor("Ravi"));
        docList.add(new Doctor("Sneha"));

        Hospital h = new Hospital("Apollo", docList);
        h.showDoctors();
    }
}
```

Output:

```
Hospital: Apollo
```

- Dr. Anjali
- Dr. Ravi
- Dr. Sneha