Practical work #6. Class relationships in Java

Lectutrer: T.Kuchkorov

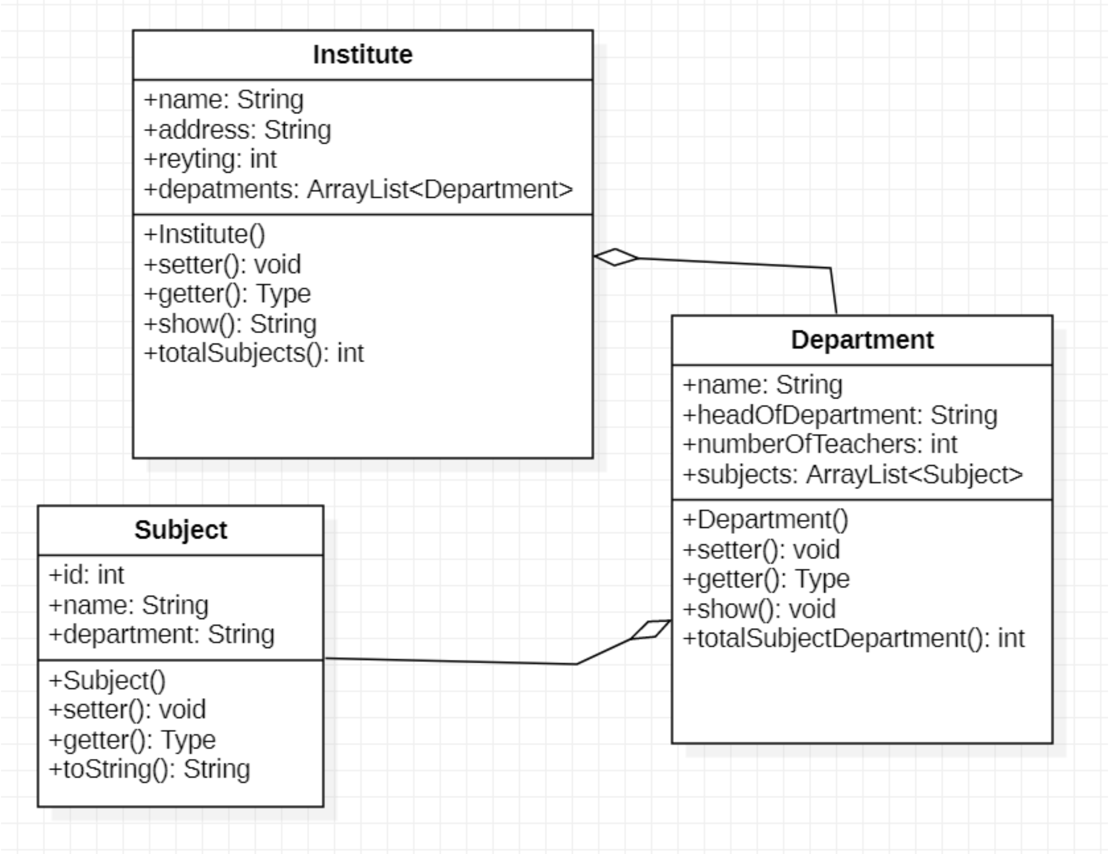
OOP | 30.03.2022

Establishing relationships between classes in Java programming language. Aggregation and Composition.

In this practical work, students will aquire the ability to use class relationships such as aggregation, and composition in the Java programming language. Student will use weak and strong connectins beetween classes regarding given tasks.

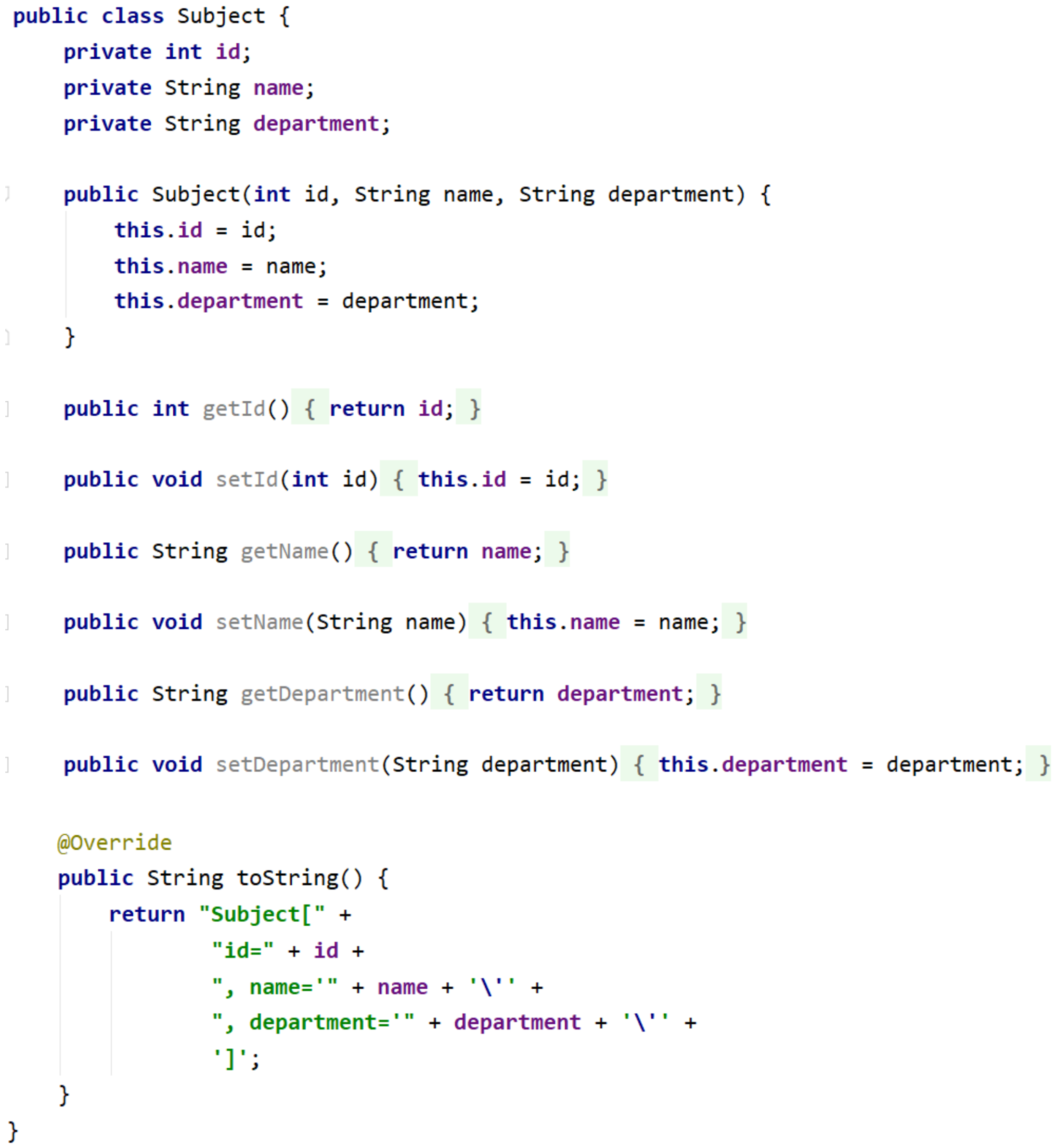
Classwork Example.

Please create classes which are given following UML diagram:



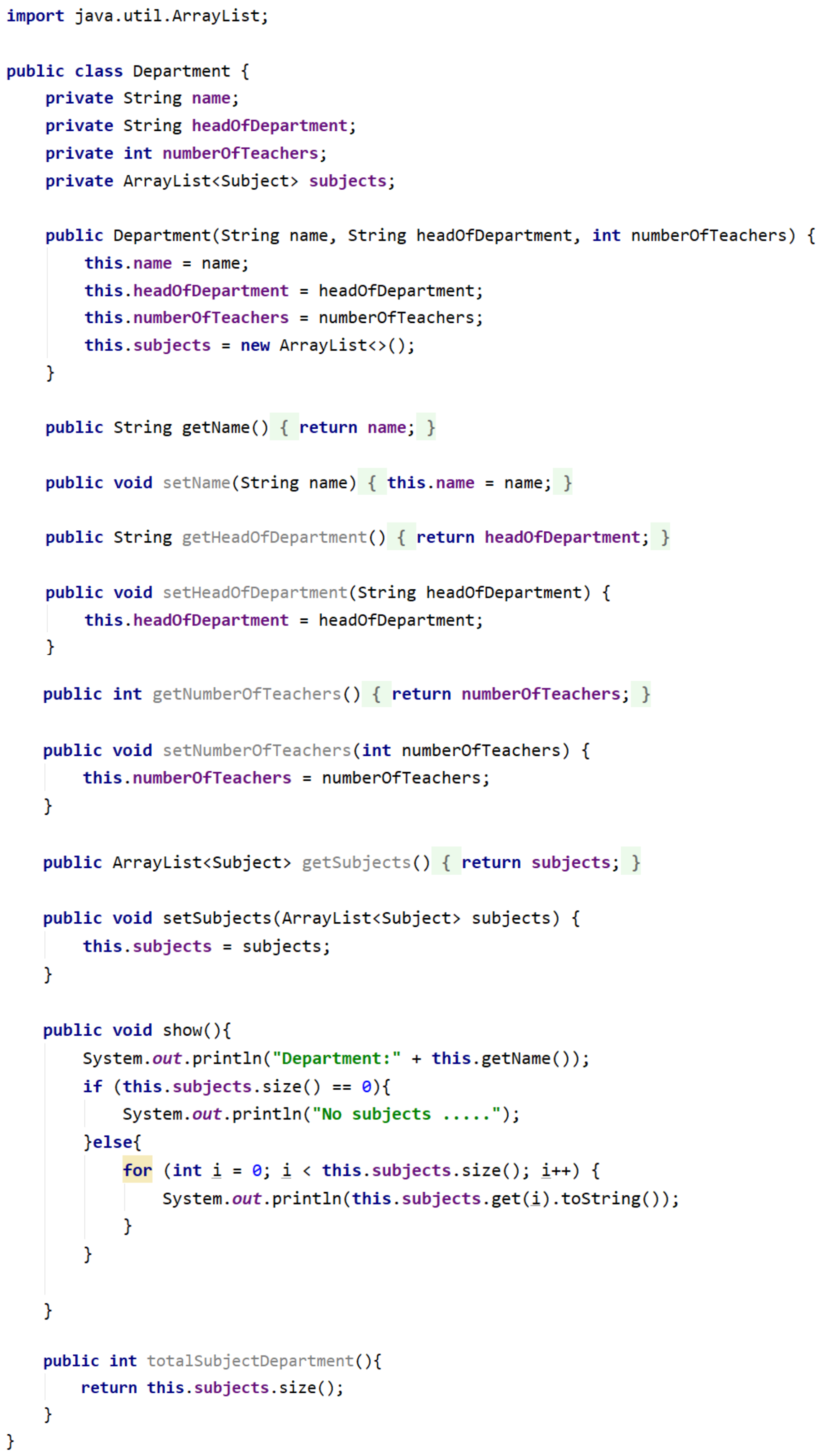
In this diagram, Subject class is an attribute of Department class (Department has-a Subject), it means that several subjects correspond to one department. Department class is an attribute of Institute class, so this means several department correspond to one institute.

Class nmae: **Subject**



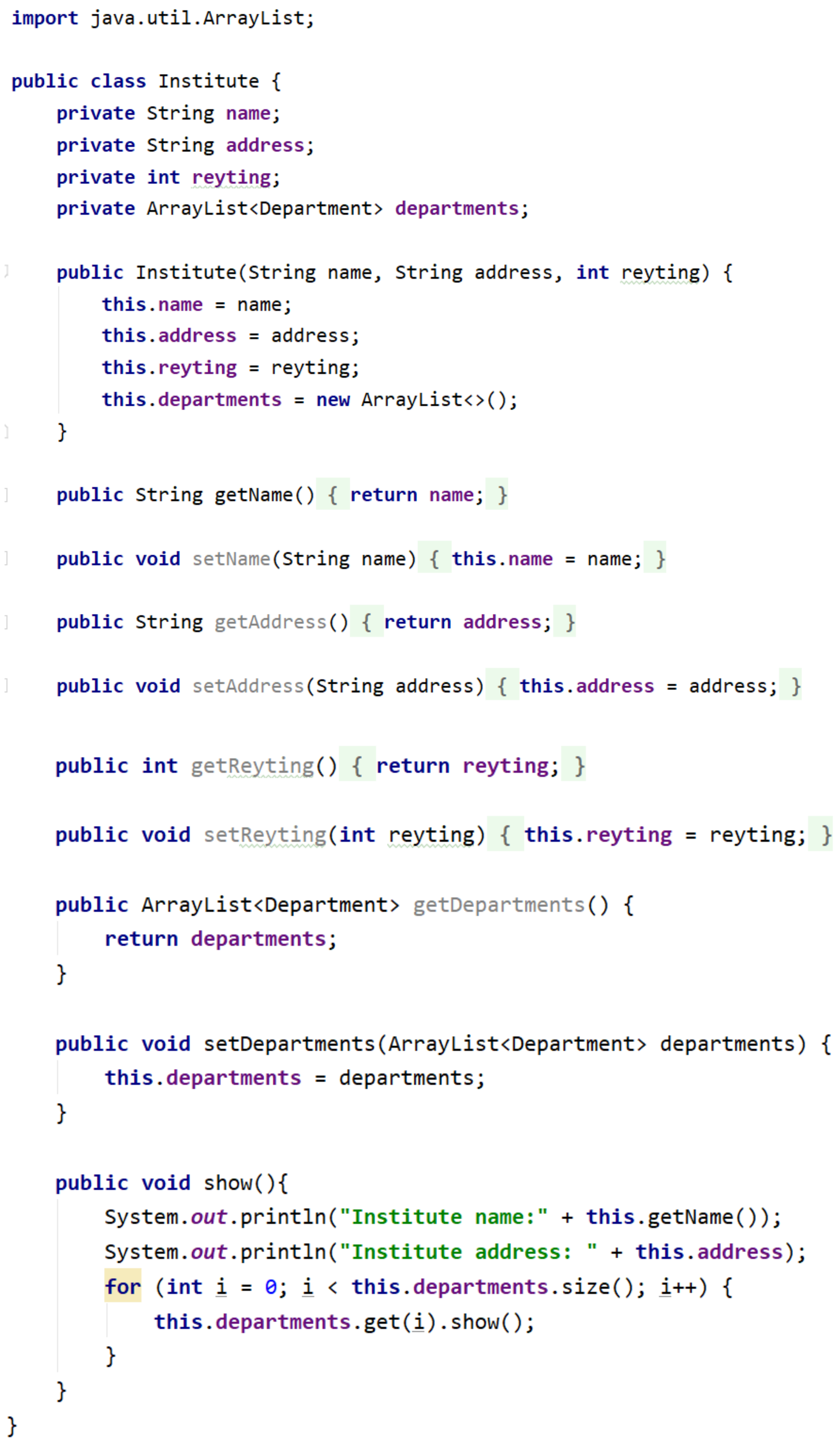
```
public class Subject {
    private int id;
    private String name;
    private String department;

    public Subject(int id, String name, String department) {
        this.id = id;
        this.name = name;
        this.department = department;
    }

    public int getId() { return id; }

    public void setId(int id) { this.id = id; }

    public String getName() { return name; }

    public void setName(String name) { this.name = name; }

    public String getDepartment() { return department; }

    public void setDepartment(String department) { this.department = department; }

    @Override
    public String toString() {
        return "Subject[" +
                "id=" + id +
                ", name='" + name + '\'' +
                ", department='" + department + '\'' +
                ']';
    }
}
```

Class name: **Department**



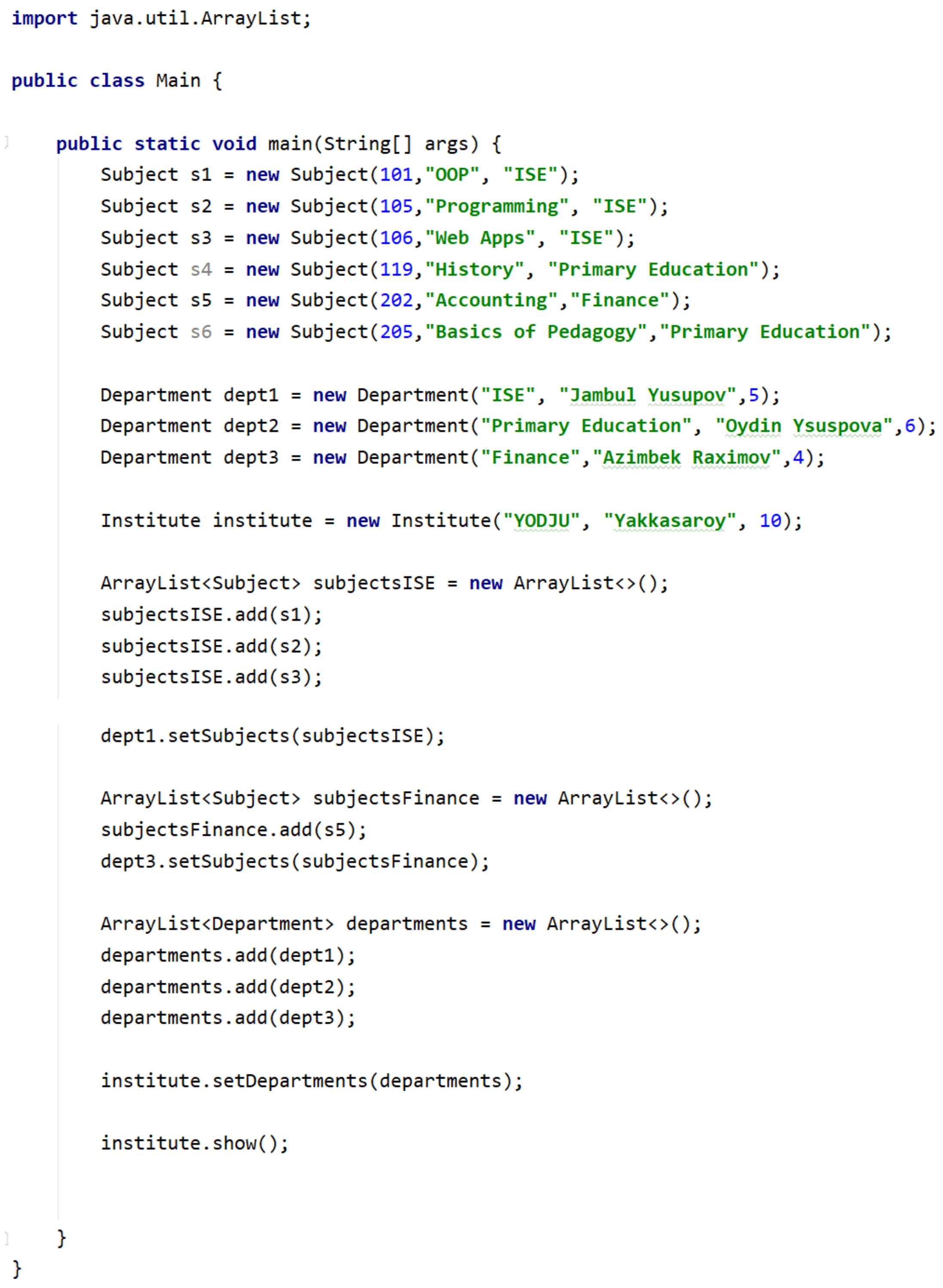
```java
import java.util.ArrayList;

public class Department {
    private String name;
    private String headOfDepartment;
    private int numberOfTeachers;
    private ArrayList<Subject> subjects;

    public Department(String name, String headOfDepartment, int numberOfTeachers) {
        this.name = name;
        this.headOfDepartment = headOfDepartment;
        this.numberOfTeachers = numberOfTeachers;
        this.subjects = new ArrayList<>();
    }

    public String getName() { return name; }

    public void setName(String name) { this.name = name; }

    public String getHeadOfDepartment() { return headOfDepartment; }

    public void setHeadOfDepartment(String headOfDepartment) {
        this.headOfDepartment = headOfDepartment;
    }

    public int getNumberOfTeachers() { return numberOfTeachers; }

    public void setNumberOfTeachers(int numberOfTeachers) {
        this.numberOfTeachers = numberOfTeachers;
    }

    public ArrayList<Subject> getSubjects() { return subjects; }

    public void setSubjects(ArrayList<Subject> subjects) {
        this.subjects = subjects;
    }

    public void show(){
        System.out.println("Department:" + this.getName());
        if (this.subjects.size() == 0){
            System.out.println("No subjects .....");
        }else{
            for (int i = 0; i < this.subjects.size(); i++) {
                System.out.println(this.subjects.get(i).toString());
            }
        }

    }

    public int totalSubjectDepartment(){
        return this.subjects.size();
    }
}
```

Class name: **Institute**



```java
import java.util.ArrayList;

public class Institute {
    private String name;
    private String address;
    private int reyting;
    private ArrayList<Department> departments;

    public Institute(String name, String address, int reyting) {
        this.name = name;
        this.address = address;
        this.reyting = reyting;
        this.departments = new ArrayList<>();
    }

    public String getName() { return name; }

    public void setName(String name) { this.name = name; }

    public String getAddress() { return address; }

    public void setAddress(String address) { this.address = address; }

    public int getReyting() { return reyting; }

    public void setReyting(int reyting) { this.reyting = reyting; }

    public ArrayList<Department> getDepartments() {
        return departments;
    }

    public void setDepartments(ArrayList<Department> departments) {
        this.departments = departments;
    }

    public void show(){
        System.out.println("Institute name:" + this.getName());
        System.out.println("Institute address: " + this.address);
        for (int i = 0; i < this.departments.size(); i++) {
            this.departments.get(i).show();
        }
    }
}
```

Main.java



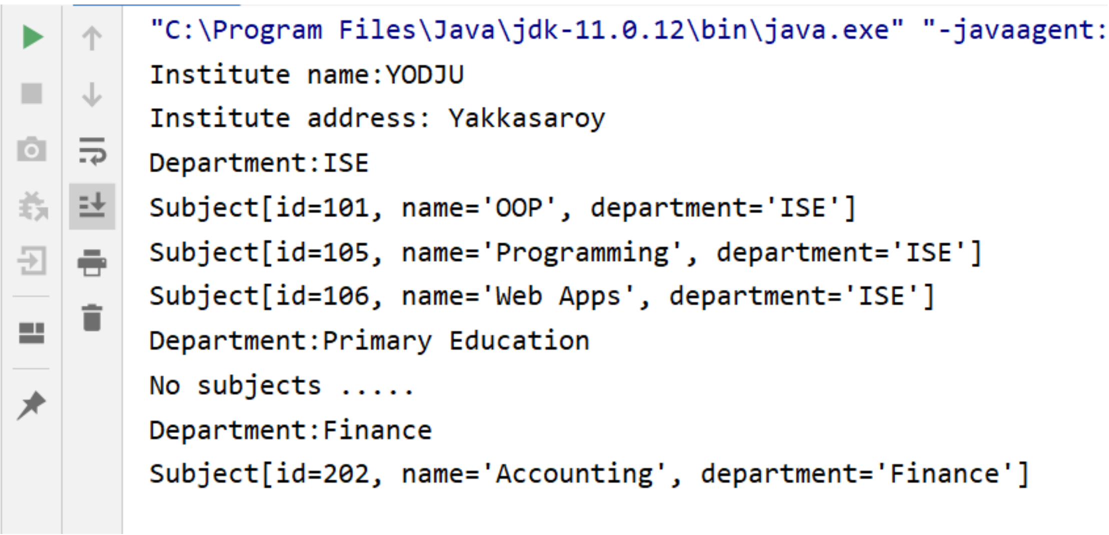
```
import java.util.ArrayList;

public class Main {

    public static void main(String[] args) {
        Subject s1 = new Subject(101,"OOP", "ISE");
        Subject s2 = new Subject(105,"Programming", "ISE");
        Subject s3 = new Subject(106,"Web Apps", "ISE");
        Subject s4 = new Subject(119,"History", "Primary Education");
        Subject s5 = new Subject(202,"Accounting","Finance");
        Subject s6 = new Subject(205,"Basics of Pedagogy","Primary Education");

        Department dept1 = new Department("ISE", "Jambul Yusupov",5);
        Department dept2 = new Department("Primary Education", "Oydin Ysuspova",6);
        Department dept3 = new Department("Finance","Azimbek Raximov",4);

        Institute institute = new Institute("YODJU", "Yakkasaroy", 10);

        ArrayList<Subject> subjectsISE = new ArrayList<>();
        subjectsISE.add(s1);
        subjectsISE.add(s2);
        subjectsISE.add(s3);

        dept1.setSubjects(subjectsISE);

        ArrayList<Subject> subjectsFinance = new ArrayList<>();
        subjectsFinance.add(s5);
        dept3.setSubjects(subjectsFinance);

        ArrayList<Department> departments = new ArrayList<>();
        departments.add(dept1);
        departments.add(dept2);
        departments.add(dept3);

        institute.setDepartments(departments);

        institute.show();


    }
}
```

Objects created by the classes independently. Regarding relationships (aggregation) in the UML diagram, objects will be setted using methods or constructors.

For example:

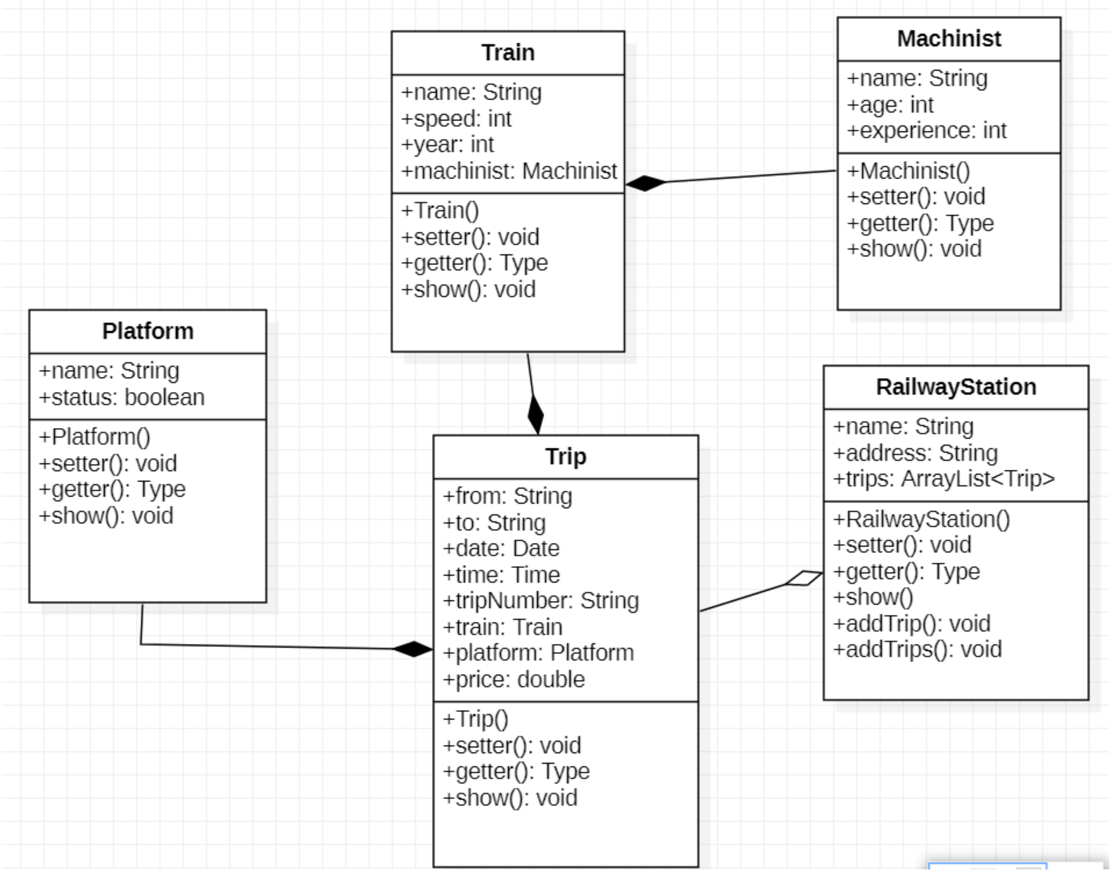
dept1.setSubjects(subjectsISE) and **institute.setDepartments(departments)**

Program result:



```
"C:\Program Files\Java\jdk-11.0.12\bin\java.exe" "-javaagent:
Institute name:YODJU
Institute address: Yakkasaroy
Department:ISE
Subject[id=101, name='OOP', department='ISE']
Subject[id=105, name='Programming', department='ISE']
Subject[id=106, name='Web Apps', department='ISE']
Department:Primary Education
No subjects .....
Department:Finance
Subject[id=202, name='Accounting', department='Finance']
```

Create following classes according to the given parameters and relationships.



Machinist part-of Train

Platform part-of Trip

Train part-of Trip

RailwayStation has-a Trip

Task:

1. Create given classes
2. Connect objects based on type of relatuonships
3. Create a program to provide information about all trips at the station using **railwayStation.show()** method