POLITEKNIK STATISTIKA STIS

For Better Official Statistics

Unified Modelling Languge (UML) and Class Diagram

by: Wa Ode Zuhayeni M

Jakarta, March 2022

Outline

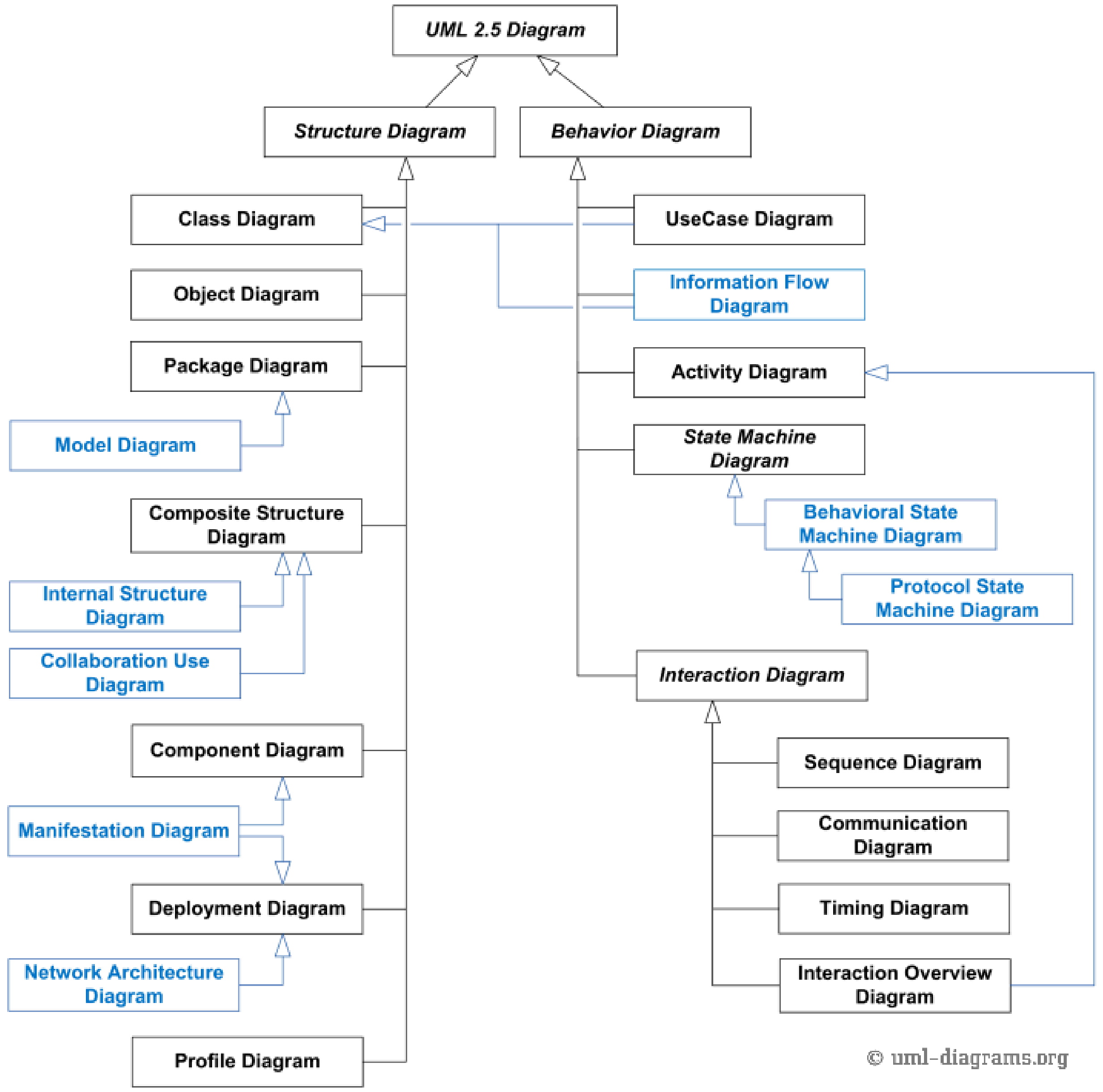
- Basic concept of UML
- Basic concept of Class Diagram
- Class Diagram Notations
- Study Case

Basic Concept of UML

- **UML (Unified Modelling Language)** is a graphical language that is suit-able to express software or system requirements, architecture, and design.
- We can use UML to communicate with other developers, clients, and increasingly, with automated tools that generate parts of our system.
- The newest version is UML 2.5 (https://www.uml-diagrams.org/uml-25-diagrams.html)

"No, it's not a pie chart; it's just a corn chip that got scanned into the document."

UML 2.5 Diagrams Overview



Note, items shown in blue are **not** part of official UML 2.5 taxonomy of diagrams.

- **Structure Diagram** : show the building blocks of the system – features that don't change with time.
- **Behavior Diagram** : show how the system responds to requests or otherwise evolves over time.

Now, We will focus on Class Diagram.

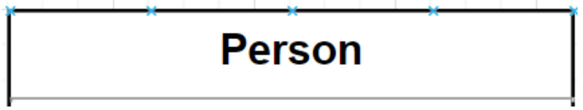
Basic Concept of Class Diagram

- **Class diagram** is UML structure diagram which shows structure of the designed system at the level pf classes and interfaces, show their features, contraints, and relationships – associations, generalizations, dependencies, etc.

Class Diagram - Notations

- **Class**

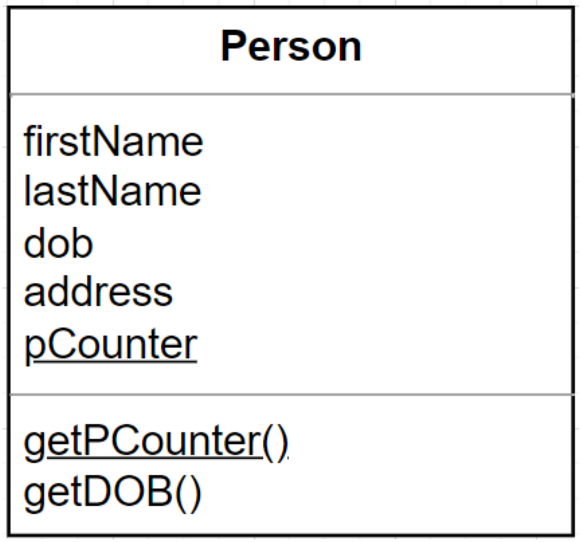
Name of the Class



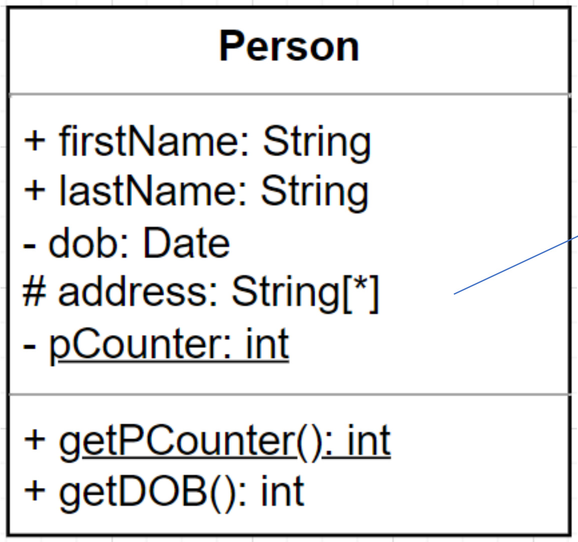
(a)

Attributes of the Class

Class variable/ operation (Static)



(b)



(c)

Multiplicity

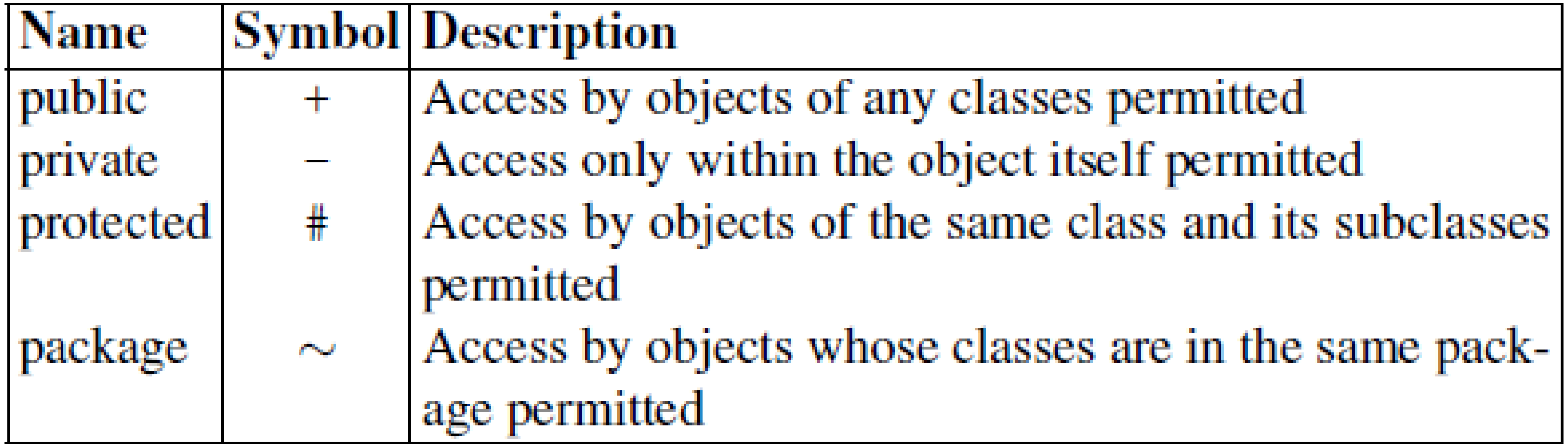
The multiplicity of an attribute indicates how many values an attribute Multiplicity can contain.

Operations of the Class

Class Diagram - Notations

- **Class**

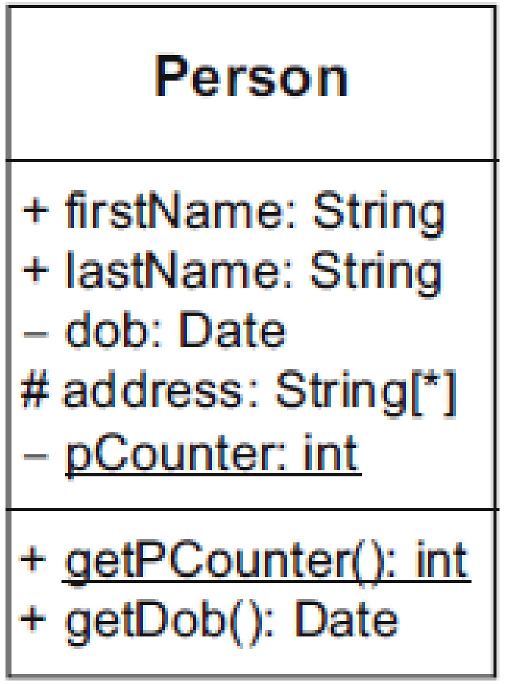
Visibilities are used to realize information hiding, an important concept in computing.

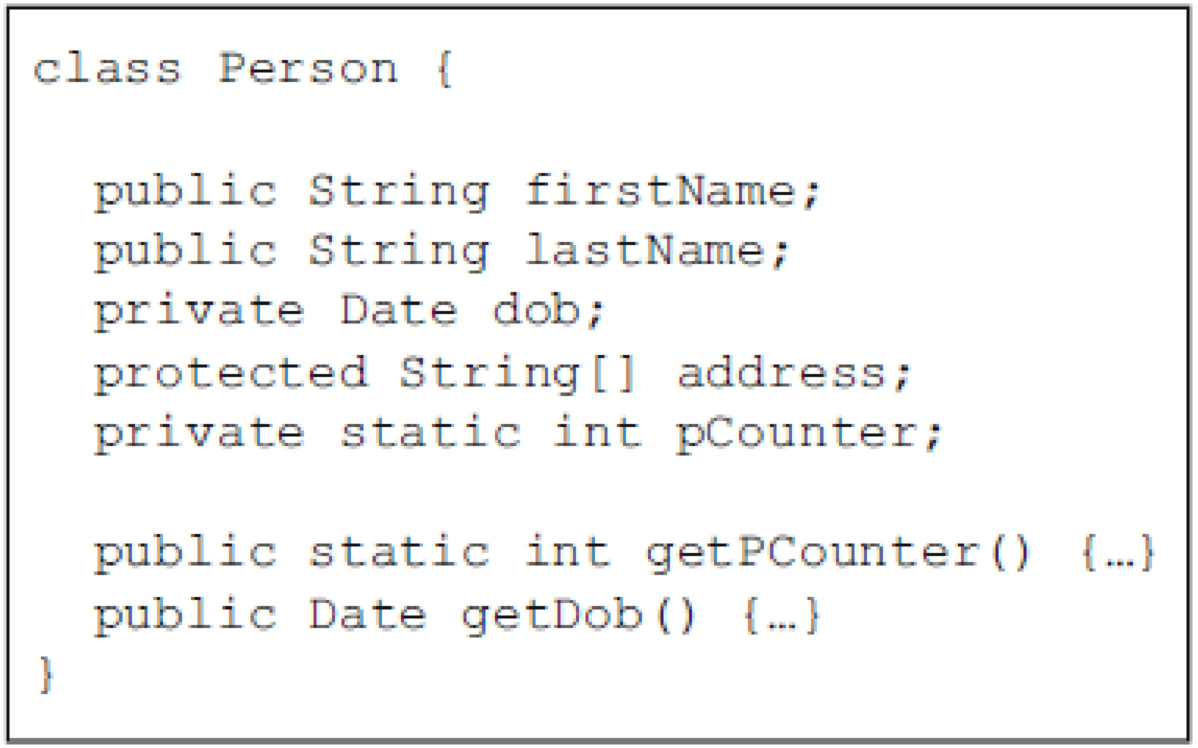


Name	Symbol	Description
public	+	Access by objects of any classes permitted
private	−	Access only within the object itself permitted
protected	#	Access by objects of the same class and its subclasses permitted
package	~	Access by objects whose classes are in the same package permitted

Class Diagram - Notations

- **Class**





```
class Person {

   public String firstName;
   public String lastName;
   private Date dob;
   protected String[] address;
   private static int pCounter;

   public static int getPCounter() {…}
   public Date getDob() {…}
}
```

Class Diagram - Notations

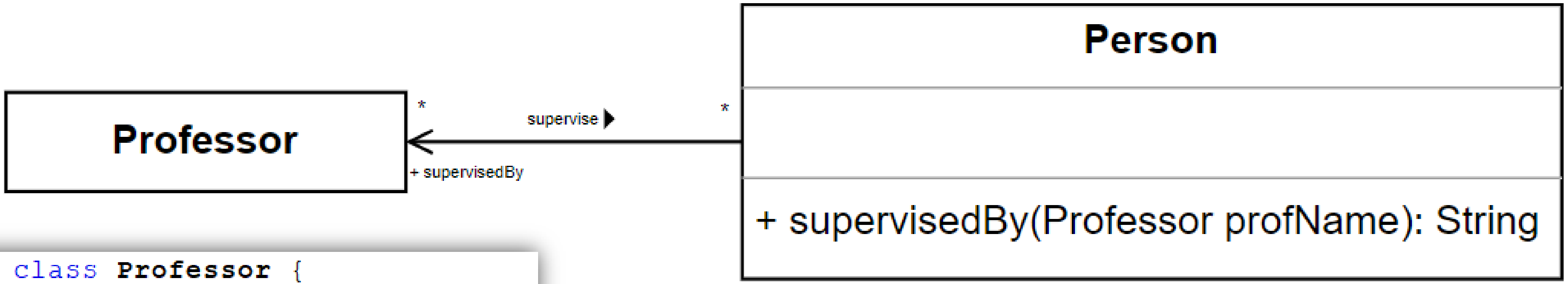
- **Multiplicities**
  - ❑ 1 (a student must have exactly one supervisor)
  - ❑ 0 .. 1 (a student may or may not join an extracurricular activities)
  - ❑ * (a lecturer cannot have a supervised student and there is no upper limit to the number of supervised student)

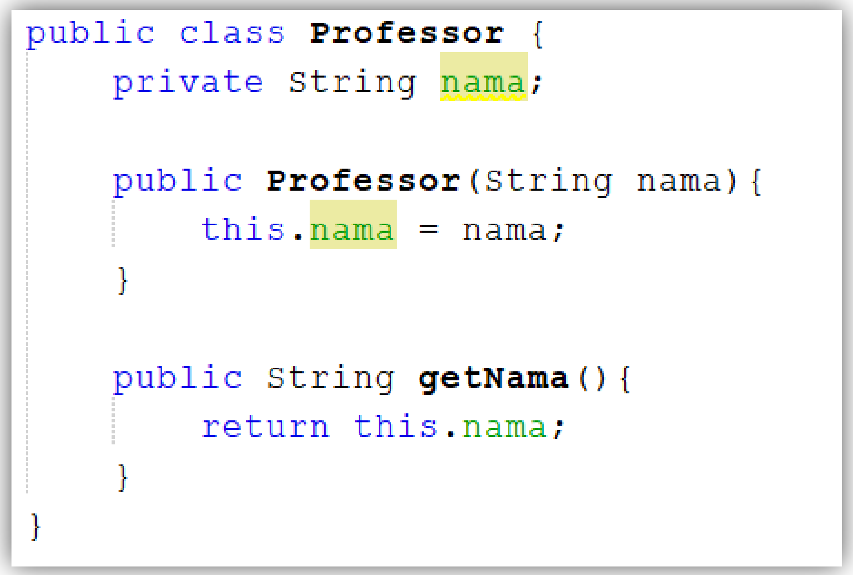
Class Diagram - Notations

- **Associations**
  - ❑ Associations between classes model possible relationships, known as links, between instances of the classes.
  - ❑ They describe which classes are potential communication partners.
  - ❑ If their attributes and operations have the corresponding visibilities, the communication partners can access each other's attributes and operations.
  - ❑ The edge can be labeled with the name of the association optionally followed by the reading direction, a small, black triangle.
  - ❑ If the edge is directed, that is, at least one of the two ends has an open arrowhead, navigation from an object to its partner object is possible.
  - ❑ In simple terms, navigability indicates that an object knows its partner objects and can therefore access their visible attributes and operations.
  - ❑ The navigation direction has nothing to do with the reading direction

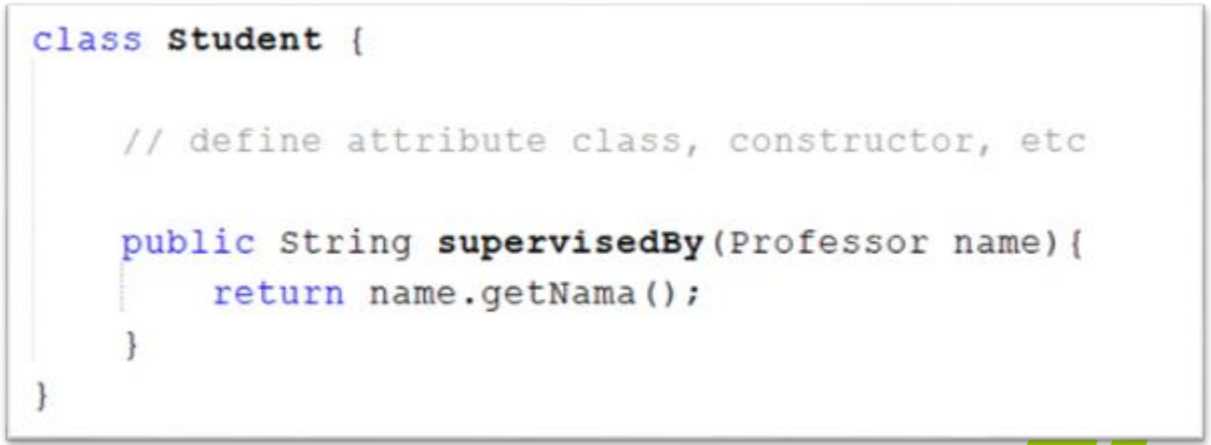
Class Diagram - Notations

- **Associations**
  - ❖ A Professor can supervise for no, one, or more than one student
  - ❖ A student can have no Professor if she/he is not in the final year





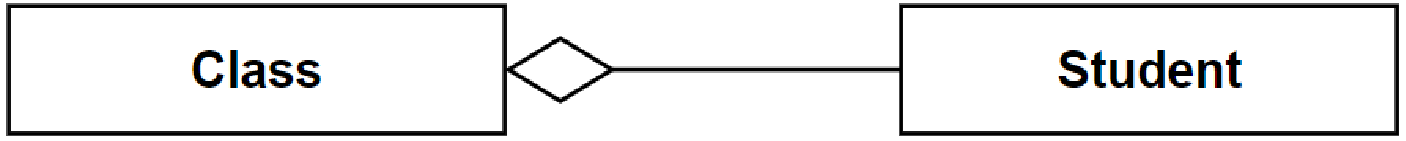
```
public class Professor {
    private String nama;

    public Professor(String nama){
        this.nama = nama;
    }

    public String getNama(){
        return this.nama;
    }
}
```

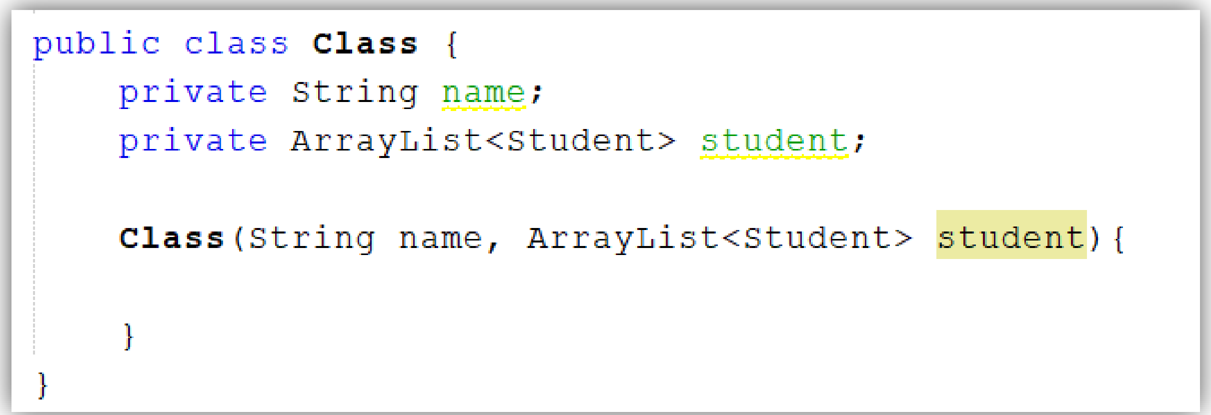


```
class Student {

    // define attribute class, constructor, etc

    public String supervisedBy(Professor name){
        return name.getNama();
    }
}
```

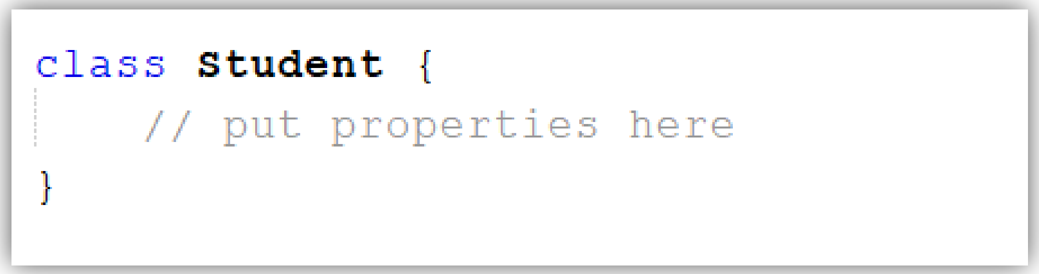
Class Diagram - Notations

- **Aggregations**
  - ❖ A class consist of any number of students





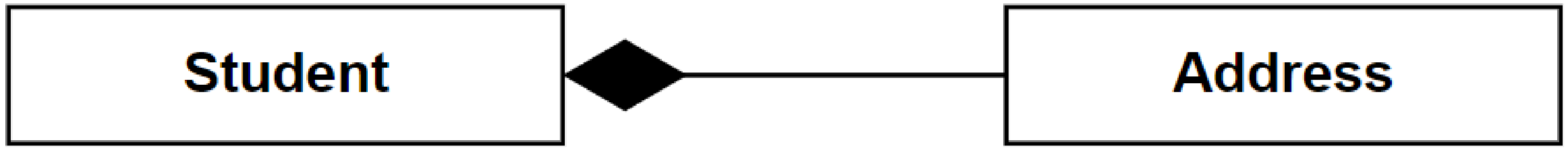
```
public class Class {
    private String name;
    private ArrayList<Student> student;

    Class(String name, ArrayList<Student> student){

    }
}
```

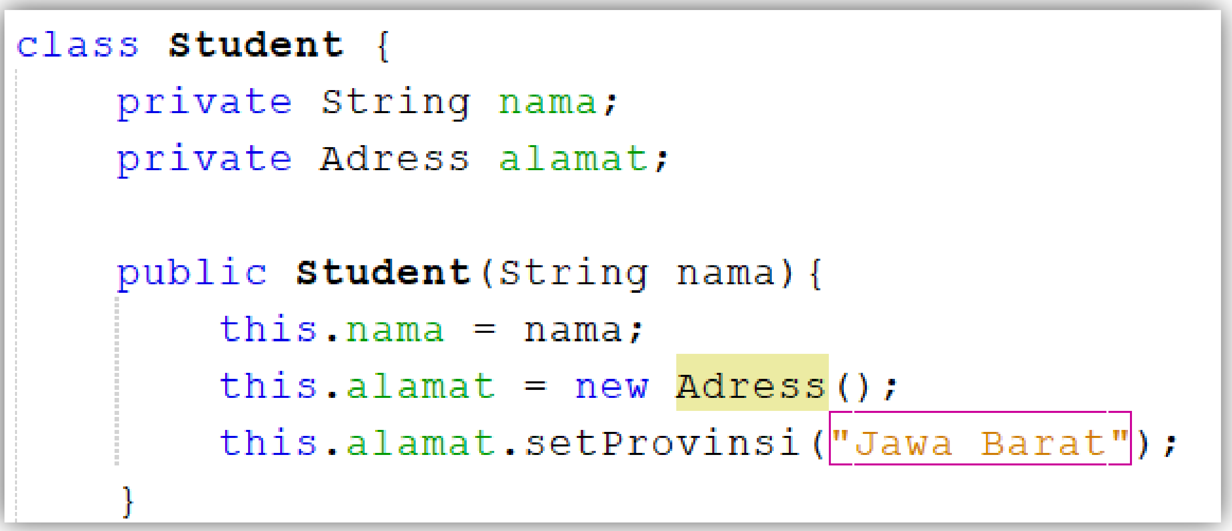


```
class Student {
    // put properties here
}
```

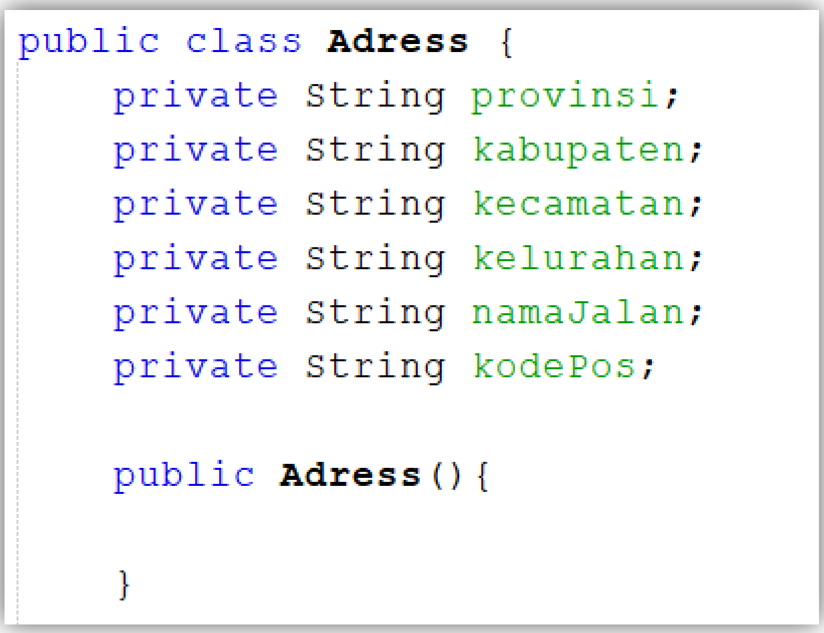
Class Diagram - Notations

- **Compositions**
  - ❖ A student must have an address that will be input in the system
  - ❖ Address consist of province, regency, and so on.





```
class Student {
    private String nama;
    private Adress alamat;

    public Student(String nama){
        this.nama = nama;
        this.alamat = new Adress();
        this.alamat.setProvinsi("Jawa Barat");
    }
```



```
public class Adress {
    private String provinsi;
    private String kabupaten;
    private String kecamatan;
    private String kelurahan;
    private String namaJalan;
    private String kodePos;

    public Adress(){

    }
```

Class Diagram - Notations

- **Generalizations**
  - ❑ We can use a generalization relationship to highlight commonalities between classes, meaning that we no longer have to define these common characteristics multiple times.
  - ❑ Conversely, we can use the generalization to derive more specific classes from existing classes.
  - ❑ The generalization relationship expresses that the characteristics (at superclass to subclass atributtes and operations) and associations that are specified for a general class (superclass) are passed on to its subclasses.
  - ❑ Generalization relationship is also referred to as inheritance.

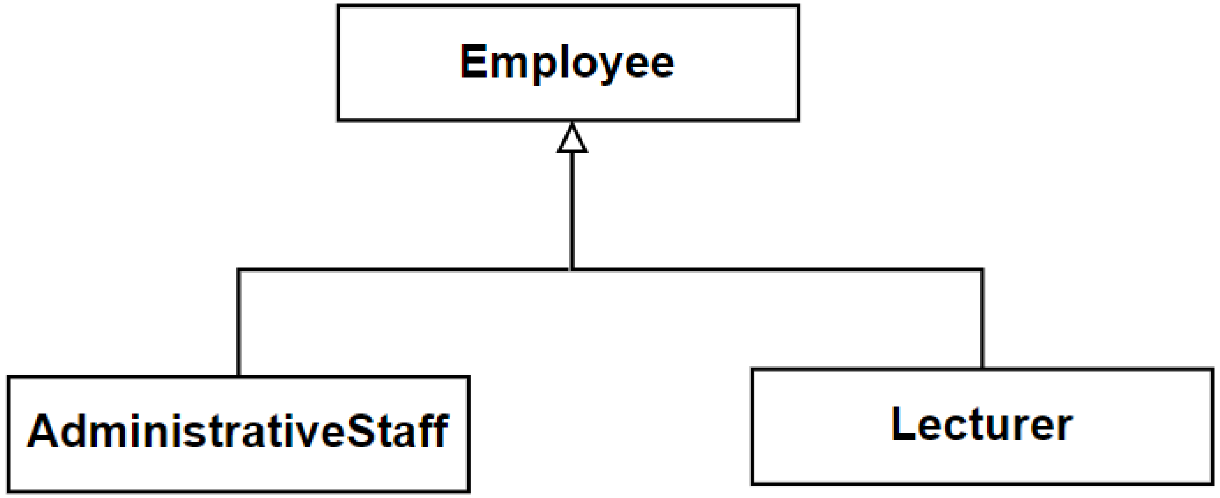
Class Diagram - Notations

- **Inheritance**
  - ❖ A university consists of employees.
  - ❖ Employee can be an administrative staff or a lecturer
  - ❖ Administrative staff is responsible for correspondence and other administration
  - ❖ A Lecturer teaches several subject

```
public class Employee {

}
```

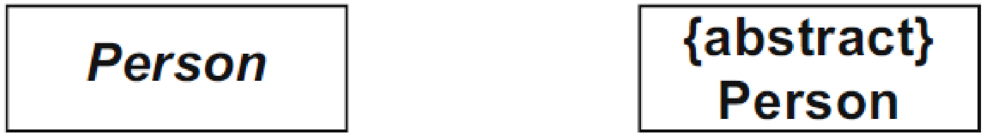
```
public class AdministrativeStaff extends Employee {

}
```

```
public class Lecture extends Employee{

}
```

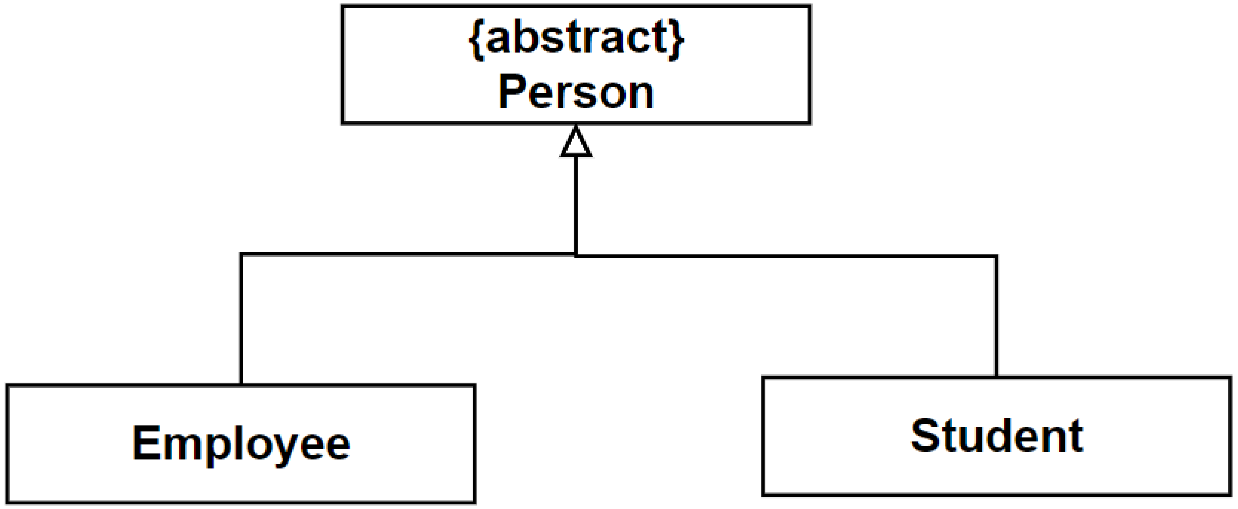


Class Diagram - Notations

- **Abstract Class**

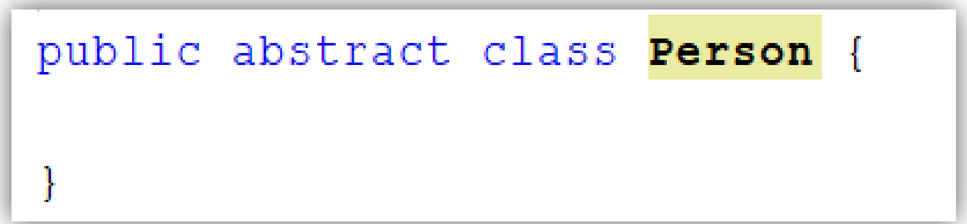


(a)



(b)

- ❖ A university consists of employees and students.
- ❖ Employees and students have common characteristics.
- ❖ Employee and student can be generalized by Person.
- ❖ There is no need to create object Person.



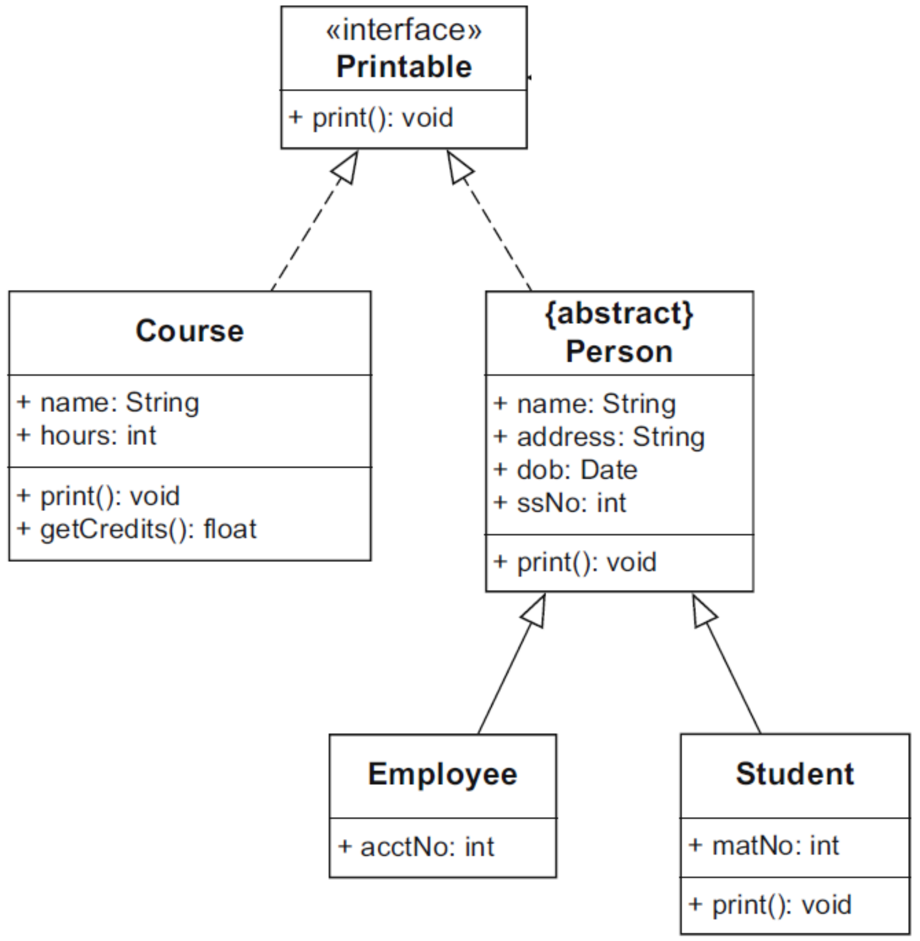
```
public abstract class Person {

}
```

Class Diagram - Notations

- **Interface**
  - ❑ An interface is denoted like a class but with the additional keyword «interface» before the name.
  - ❑ A dashed inheritance arrow with a hollow, triangular arrowhead from a class to an interface signifies that this class implements the interface.
  - ❑ Case:
    - ❖ The list of courses contained in a university needs to be printed in a certain format for promoting their program.
    - ❖ The university also need to print a list of active students and employees for monitoring and evaluation.

Class Diagram - Notations

- **Interface**



```
public interface Printable {

}
```

Study Case: Information System of a University

- ❑ A university consists of multiple faculties which are composed of university various institutes. Each faculty and each institute has a name. An address is known for each institute.
- ❑ Each faculty is led by a dean, who is an employee of the university.
- ❑ The total number of employees is known. Employees have a social security number, a name, and an e-mail address. All employees have same obligation, namely to take the attendance. There is a distinction between research and administrative personnel.
- ❑ Calculation of attendance percentages are different for both research and administrative personnel. Administrative personnel: based on routine attendance per day. Research personnel: based on the number of teaching hours
- ❑ Research associates are assigned to at least one institute. The field of study of each research associate is known. Furthermore, research associates can be involved in projects for a certain number of hours, and the name, starting date, and end date of the projects are known. Some research associates teach courses. They are called lecturers.
- ❑ Courses have a unique number (ID), a name, and a weekly duration in hours.

Terima Kasih