UML - Class Diagrams
What is UML Diagram?

- Unified Modeling Language (UML)
- Standardized general-purpose modeling language in the field of object-oriented software engineering.
- The standard is managed, and was created, by the Object Management Group.
UML 2.2 has 14 types of diagrams divided into two categories.

- Seven diagram types represent structural information.
- Seven represent general types of behavior.
UML Diagram Types

Diagram

Structure Diagram

Class Diagram

Profile Diagram

Component Diagram

Composite Structure Diagram

Deployment Diagram

Object Diagram

Package Diagram

Behaviour Diagram

Activity Diagram

Use Case Diagram

Interaction Diagram

State Machine Diagram

Notation: UML

Sequence Diagram

Communication Diagram

Interaction Overview Diagram

Timing Diagram
What is class diagram?

- It is a type of **static structure diagram** that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes.

```plaintext
BankAccount
owner : String
balance : Dollars = 0

deposit (amount : Dollars)
withdrawl (amount : Dollars)
```
# Class Diagram Models

<table>
<thead>
<tr>
<th>Class Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attributes</td>
</tr>
<tr>
<td>Methods</td>
</tr>
</tbody>
</table>
Example:
CD Models (Class and Attribute)

```java
public class Person {
    private String name;
}
```

**Modifier**

+ : Public
- : Private
# : Protected
~ : Package
Example 2:
CD Models (Class, Attributes, Method)

```java
public class MyCar{
    public String Color;
    public int yearOfProduction;
    public void printMyCar(){
        System.out.println("Color: " + Color);
        System.out.println("Year Of production: " + yearOfProduction);
    }
}
```

MyCar

- + Color : String
- + yearOfProduction : int
- + printMyCar () : void
Example 3:
CD Models (Class with main method)

```java
public class MyCarInAction{
    public static void main(String[] args){
        MyCar myCar = new MyCar();

        myCar.Color = "Black";
        myCar.yearOfProduction = 2006;
        myCar.printMyCar();
    }
}
```
Exercise 1:
Create code from UML class diagram below!

```java
public class MyCar2 {
    String color;
    int yearOfProduction;

    public void printMyCar() {
        System.out.println(color + " is produced in the year of " + yearOfProduction);
    }

    public void turnOnMachine() {
        System.out.println("Machine is turned ON");
    }

    public void changeGear() {
        System.out.println("Gear is changed");
    }

    public void turnOffMachine() {
        System.out.println("Machine is turned OFF");
    }
}

public class MyCarInAction2 {
    public static void main(String[] args) {
        MyCar2 car = new MyCar2();
        car.color = "Black";
        car.yearOfProduction = 2006;
        car.turnOnMachine();
        car.changeGear();
        car.turnOffMachine();
    }
}
```
Example 4:
CD Models (Class with main constructor)

`Students`
- nim : String
- nama : String

`+ Students(nim : String, nama : String)`
  + getNim () : String
  + getNama () : String

`StudentsCard`

`+ main(String[]): void`
Example 4:
CD Models (Class with constructor) Result

```java
class Students{
    private String nim;
    private String nama;

    public Students(String nim, String nama){
        this.nim = nim;
        this.nama = nama;
    }

    public String getNim(){
        return nim;
    }

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

```java
class StudentsCard{
    public static void main(String[] args){
        Students SiAdin = new Students("A11.2012.0001","SiAdin");
        System.out.println("Student Name : " + SiAdin.getNama());
        System.out.println("Student Number : " + SiAdin.getNim());
    }
}
```
Exercise 2:
Create code from UML class diagram below!

Students2

- nim : String
- nama : String

+ Students2(nim : String)
  + setNim(String): void
  + setNama(String): void
  + getNim(): String
  + getNama(): String

StudentsCard2

+ main(String[]): void

With scanner
import java.util.scanner
Example 5:
CD Models (Inheritance)

Bicycle
- gear: int
- speed: int
+ changeGear(int): void
+ speedUp(int): void
+ applyBrake(int): void

MountainBike
- seatHeight: int
+ setHeight(int): void

MountainBikeAction
+ main(String[]): void
Example 5:
CD Models (Inheritance) Result

```java
public class Bicycle{

    //state
    private int speed ;
    private int gear ;

    // method
    public void changeGear(int newValue) {
        gear = newValue;
        System.out.println("Gear:" + gear);
    }

    public void speedUp(int increment) {
        speed = speed + increment;
        System.out.println("Speed:" + speed);
    }

    public void applyBrake(int decrement) {
        speed -= decrement;
        System.out.println("Speed:" + speed);
    }
}
```
Example 5:
CD Models (Inheritance) Result cont’d

```java
class MountainBike extends Bicycle {
    // the MountainBike subclass adds one field
    private int seatHeight;

    // the MountainBike subclass adds one method
    public void setHeight(int newValue) {
        seatHeight = newValue;
        System.out.println("Seat height:" + seatHeight);
    }
}
```

```java
class MountainBikeAction {
    public static void main(String[] args) {
        // Create a object
        MountainBike mbike = new MountainBike();

        // Calling method
        mbike.speedUp(10);
        mbike.changeGear(2);
        mbike.setHeight(20);
    }
}
```
Exercise 3:
Create code from UML class diagram below!

```
Calc

# result: double = 0
# operand1: double = 0
# operand2: double = 0

# setOperand1(double): void
# setOperand2(double): void
# getOperand1(): double
# getOperand2(): double
# addition(): double
# subtraction(): double
# multiplication(): double
# division(): double

AdvancedCalc

+ modulo(double, double): double

AdvancedCalc

+ main(String[]): void
```
Exercise 3:

Result

```
D:\DOKUMEN\UDINUS\FASILKOM\PBO\_latihan\m7>javac AdvancedCalcAction.java
D:\DOKUMEN\UDINUS\FASILKOM\PBO\_latihan\m7>java AdvancedCalcAction

-----------------------------------
---Advanced Calculator---
For Addition press 1
For Subtraction press 2
For Multiplication press 3
For Division press 4
For Modulo press 5
For Exit press 0
-----------------------------------

Enter your choice : 5

-----------------------------------

Enter First Number = 10
Second First Number = 3
10.0%3.0=1.0
-----------------------------------
```
Example 6:
CD Models (Multilevel Inheritance)

Person

- # name: String
- # address: String

# setName(String): void
# setAddress(String): void
# getName(): String
# getAddress(): String

Student

- campus: String

# setCampus(String): void
# getCampus(): String

Employee

- # totalSalary: double

# setTotalSalary(double): void
# calculateSalary(): void
# getTotalSalary(): double

Monthly

- numberOfAbsent: int
- salaryPerMonth: double

+ calculateSalary(): void
+ addAbsent(int): void

Hourly

- salaryPerHour: double

+ calculateSalary(): void
+ addHour(float): void

PersonInAction

+ main(String[]):
  void
Exercise 4:
Create code from class diagram in the previous slide

• Clue:
  - calculateSalary is polymorphism method

```java
public class Person{
}

public class Student extends Person{
}

public class Employee extends Person{
}

public class Monthly extends Employee{
}

public class Hourly extends Employee{
}
```
THANKS