Lecture 03: Object Oriented Principles



Object Oriented Design Pattern



Object Oriented Design Principles favour composition over inheritance



Object Oriented Principles – Encapsulation, Abstraction, Inheritance, Polymorphism



Classes and Objects



Encapsulation – Information hiding or data hiding

Class Cat

1. Name
2. Type
3. Age

Default constructor will be added when no constructors are added the programmer and it is also known no-args

Looks as follows:

public Cat() {

super();

this.name = null;

this.age = 0;

this.type = null;

}

Encapsulation is making the attribute private

private String name;

private int age;

private *CatType* type;

providing getters and setters with necessary validation code

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public int getAge() {

return age;

}

public void setAge(int age) {

**if (age >= 0) {**

**this.age = age;**

**} else {**

**throw new IllegalArgumentException("Age cannot be less than 0");**

**}**

}

public *CatType* getType() {

return type;

}

**Abstraction:**

The API

* **Cat(String name, int age, CatType type)**: Initializes a new Cat instance with the specified name, age, and type. Throws IllegalArgumentException if age is less than 0.

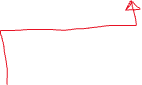
**Methods**

* **String getName()**: Returns the name of the cat.
* **void setName(String name)**: Sets the name of the cat to the specified value.
* **int getAge()**: Returns the age of the cat.
* **void setAge(int age)**: Sets the age of the cat. Throws IllegalArgumentException if age is less than 0.
* **CatType getType()**: Returns the type (enum value) of the cat.
* **String toString()**: Returns a string representation of the Cat object, including name, age, and type.

Inheritance:

Mother

Father



Child

**Multiple Inheritance**

Multiple inheritance is not supported by java – a child class has more than one parent class is known multiple inheritance

Then what type of inheritance java supports? Single Inheritance

Why doesn’t java support multiple inheritance?

Deadly Diamond Disease

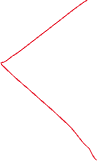
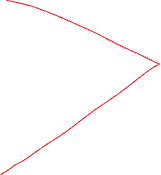
Vehicle

Vehicle::travel()



Boat

Aeroplan



Vehicle::travel() Vehicle::travel()

Seaplan

Vehicle::travel() through Aeroplan

Vehicle::travel() through Boat

Since there are two travel() inherited from Aeroplan and Boat there is an ambiguity if I create Seaplan object and call travel() there will be an ambiguity of which travel() to call

**What are the alternatives that we can use to simulate multiple inheritance in java?**

* 1. Using interface and default methods
  2. Using composition (Has a relationship)

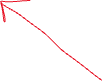
Person

Student

Lecturer

firstName

lastName



lecturerID studentID

firstName firstName

lastName lastName



deliverModule() enrolModule()