LLD 2 : OOP

Agenda

→ OOP Basic Terminology

→ Class

→ Objects

→ Fields

→ Methods

→ Members

→ State

Java for Programmers

OOP Pillars

- Abstraction
- Encapsulation
- Inheritance
- Polymorphism

Interfaces

Code

LLD

↓

Project Building

Class is in control of what can done on it

Object.doSomething() → OOP

doSomething(object) → NON OOP

→ data is vulnerable

In OOP everything revolves around

entities → Any thing that has some info / attributes AND can take some behavior

Scaler

Student

Instructor

entity.doSomething();

Class

Mentor

Problem

Video Player

Assignment

↳ list<Problem>

Problem Checker

↳

PSP number

↳

↳

PSP {
percentage
calculate
}

Problem

↳ Name

↳ Description

↳ list<SampleT>

↳ list<Samp

↳ Score

I

Entity Any element in the visualization of a software system that has attributes

OR

Causes some behaviour

II

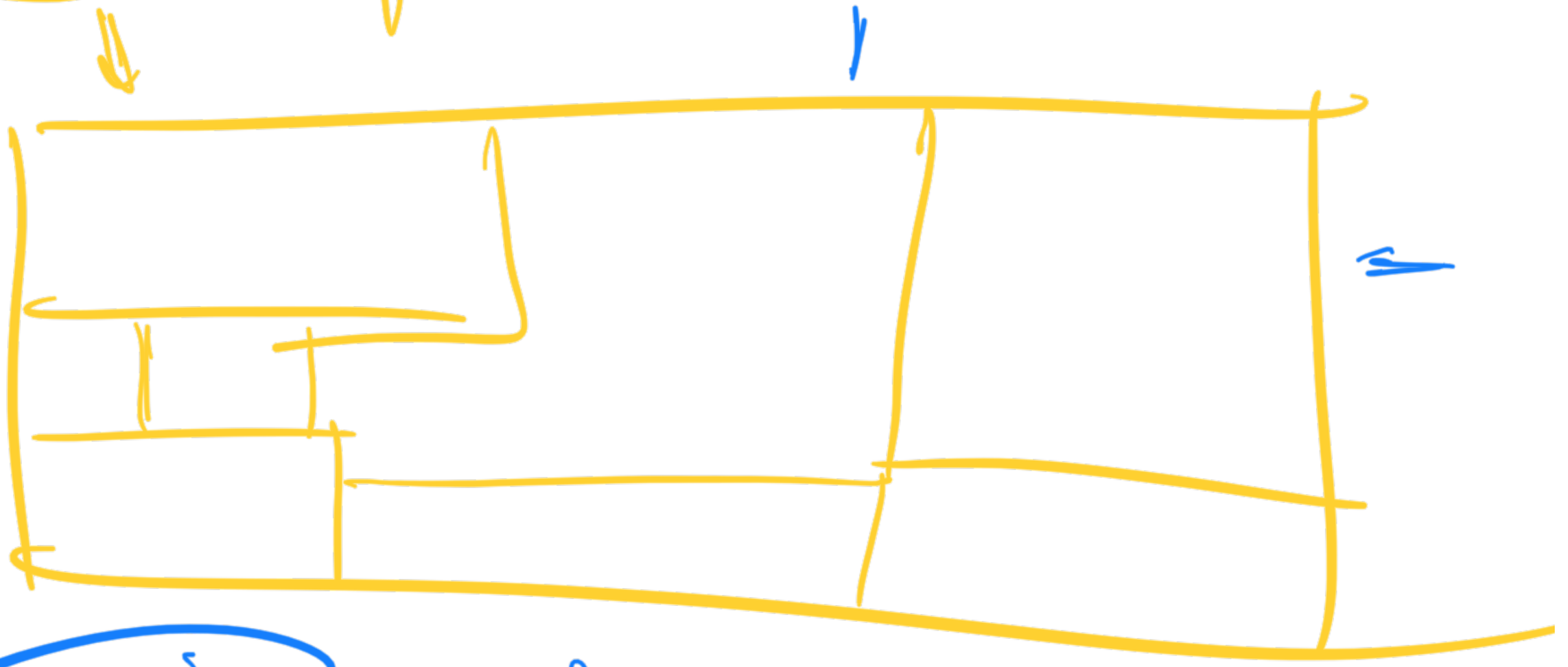
Class Blueprint of an entity

Step 1 for building house:
go to an Architect and ask them to build a design for you

based upon your requirements

from the hours

Architects



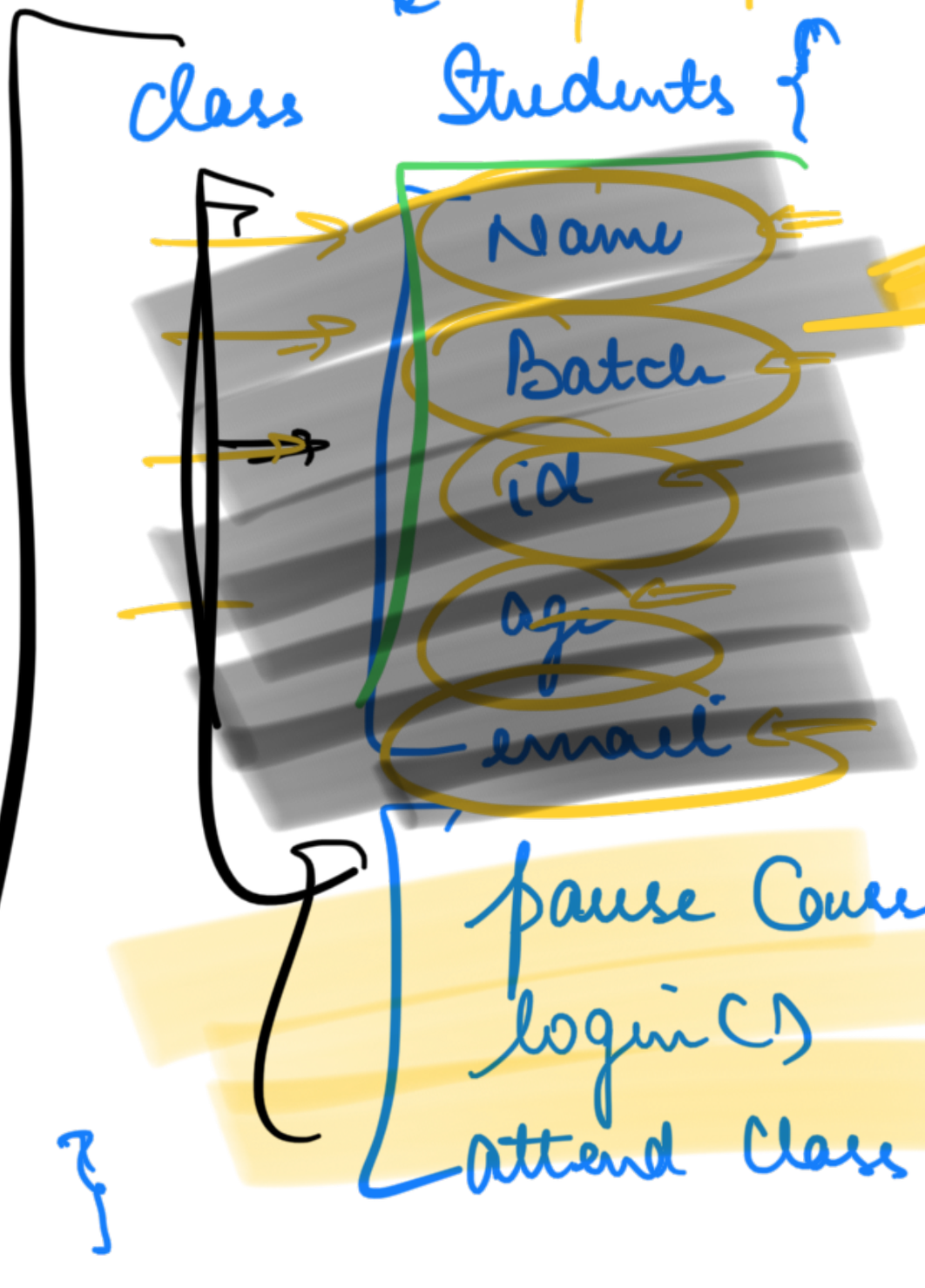
Blueprint

Blue Paper

Step2 Give blueprint to a Constructor

to get the house built.

A Blueprint of all the students in my codebase



Attributes | Fields | Properties

1 Blueprint can be used to for multiple object instances

→ methods OR behaviours

real instance of a student

Student Saurabh = new Student();

Saurabh.name =

Saurabh.batch =

Saurabh.age =

Saurabh.passCourse

Student Satyasachi = new Student();

- Name =
- age =
- batch

Each instance is indepent

Objects : Real instances of a class

Objects take memory

data that

Class

Defines the class

Members

Fields

Attributes

Properties

Methods

Behaviours

Method of a function a class

Constructor Method

A set of statements to carry some action

Entity → Class ⟹ Object

Entity – Relationship Diagram

ER

Terminology of OOP

Support

There are 3 pillars of OOP

There is 1 principle of OOP

3 ways/ concepts that support imp of OOP in any programming language

ABSTRACTION

OOP has (1 principle) that is brought into practice by 3 pillars.

Java the complete reference

1. Encapsulation
2. Inheritance
3. Polymorphism

PRINCIPLE OF OOP <=

1. ABSTRACTION =

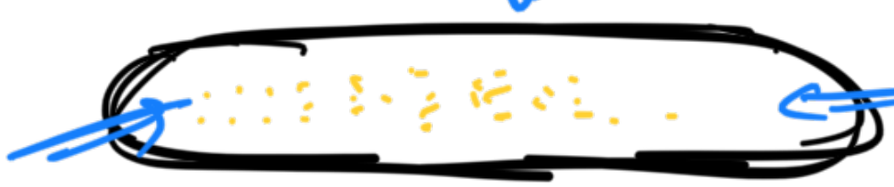
representing a
complex system
as a set of
different ideas working
together

3 pillars that bring ABSTRACTION into practice:

1. Encapsulation
2. Inheritance
3. Polymorphism

Medicine Capsules

Protect from outside



hold diff medical salts to gether

⇒ Rep every idea as a capsule.

Encapsulation means

① Holding data and methods of an entity together

② Prevention of illegitimate access to diff data members.

Access Modifier

public

Private

class Human {

private String name;

public int age;

protected

String dob;

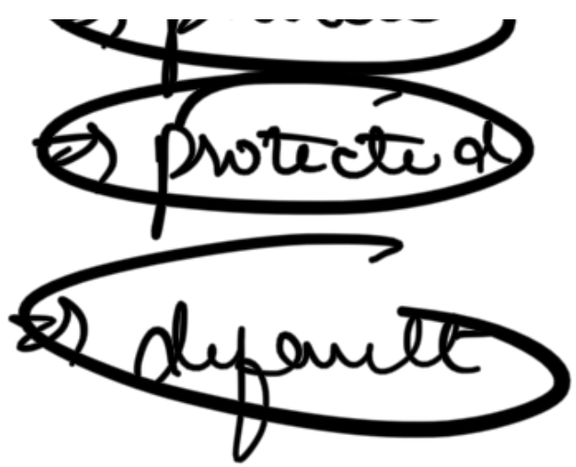
private

void walk();

public

void eat();

}



Abstraction VS Encapsulation

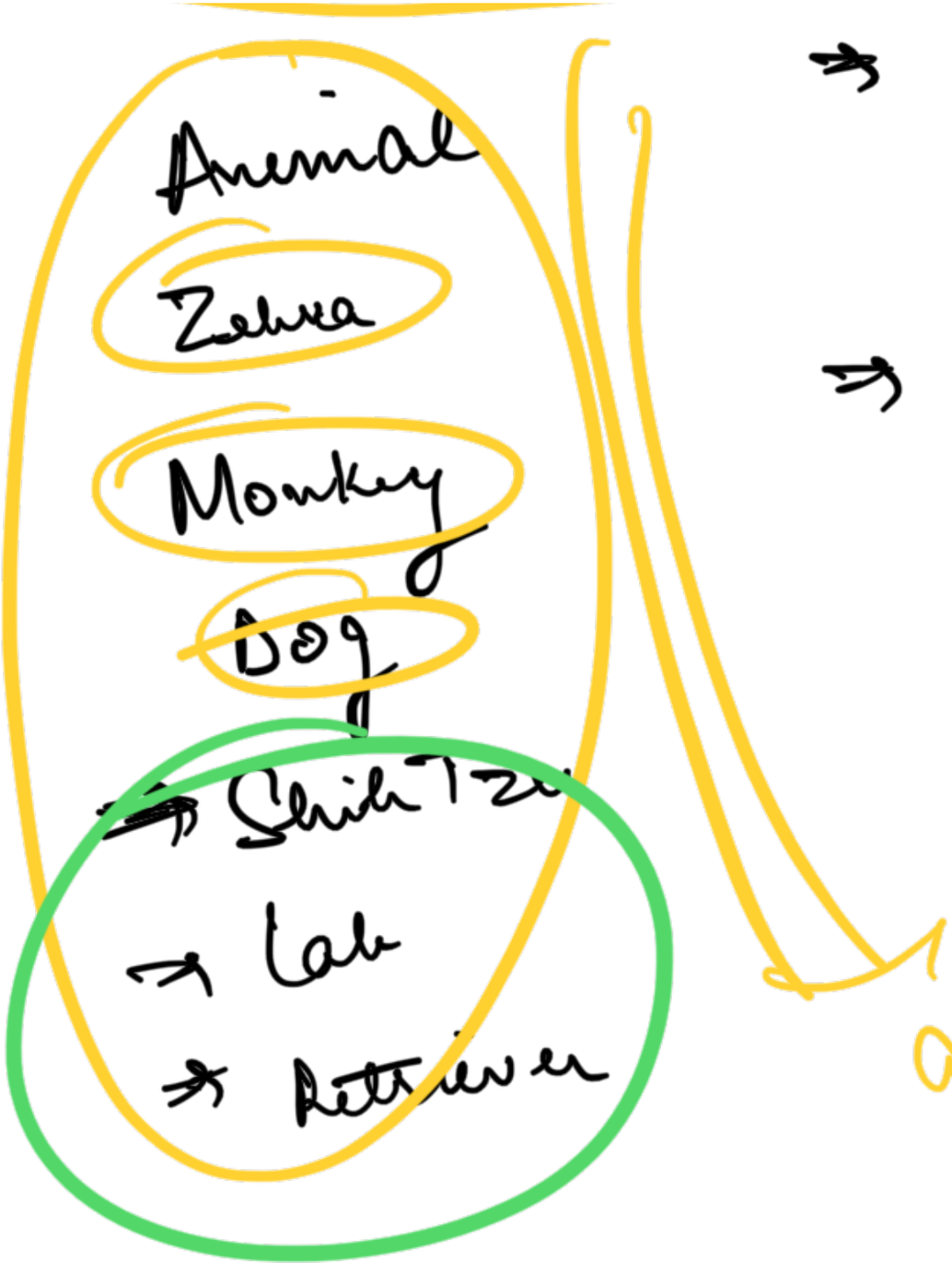
ABSTRACTION	ENCAPSULATION
→ Principle behind OOP	→ Way to support

abstraction in prog language

→ Rep a comput software system as a set of diff ideas working together to make system work

→ Holding data and behaviour together and preventing it from illegitimate uses

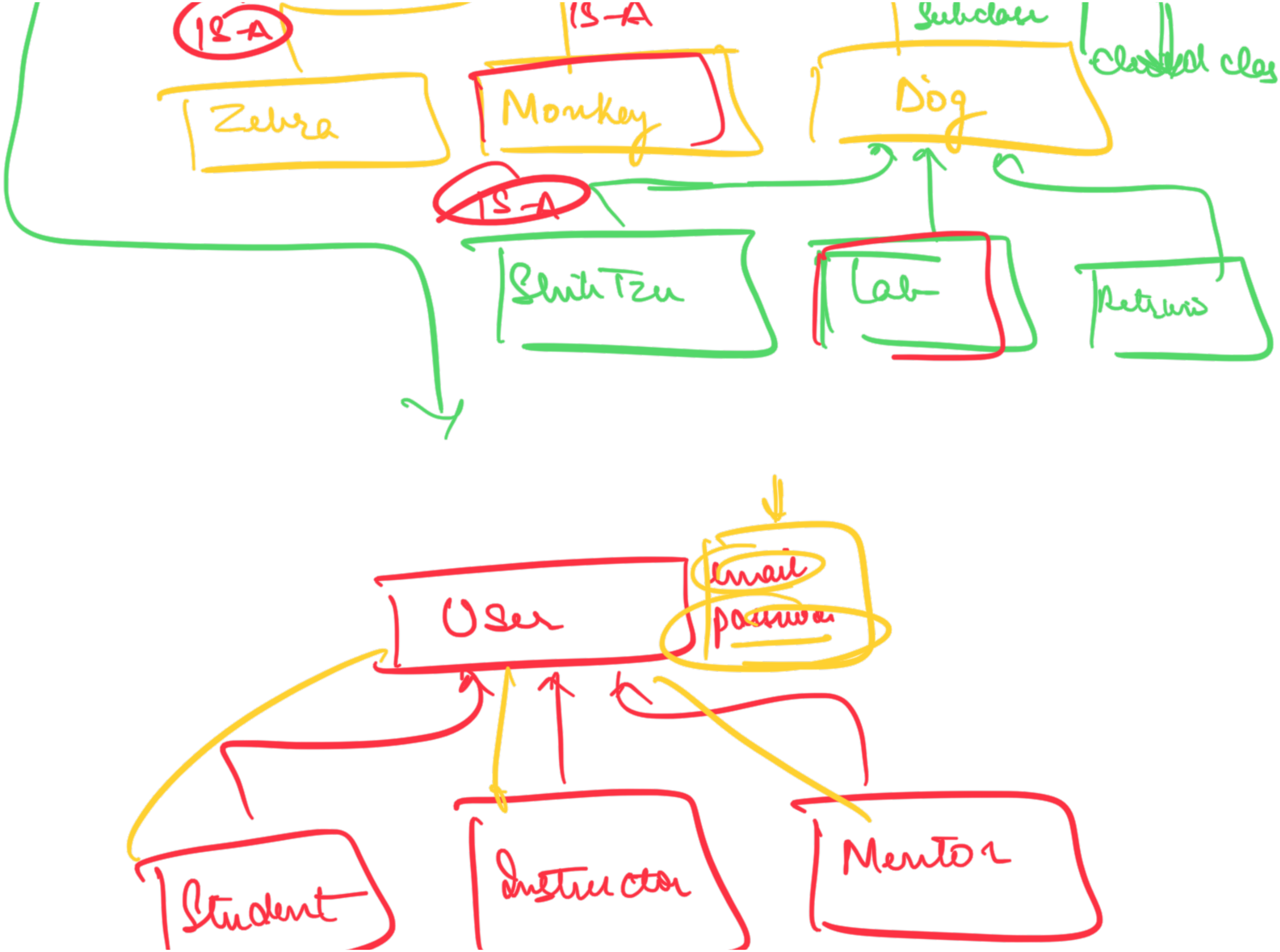
INHERITANCE → allows me to rep hierarchy in the ideas.



⇒ Some form of hierarchy in the ideas.

⇒ might share some attributes / behaviours

Allows to share attributes and methods b/w classes.


IS-A
IS-A
Subclass
Zebra
Monkey
Dog
IS-A
ShihTzu
Lab
Retriever
User
email
password
Student
Instructor
Mentor

// IS-A

class Dog extends Animal {

class Animal {

- color ;
- leg Count ;
- is Mammal ;
- age ;
- walk()
- make Sound()
- eat()

breed

Cuteness Level

bark()

wag()

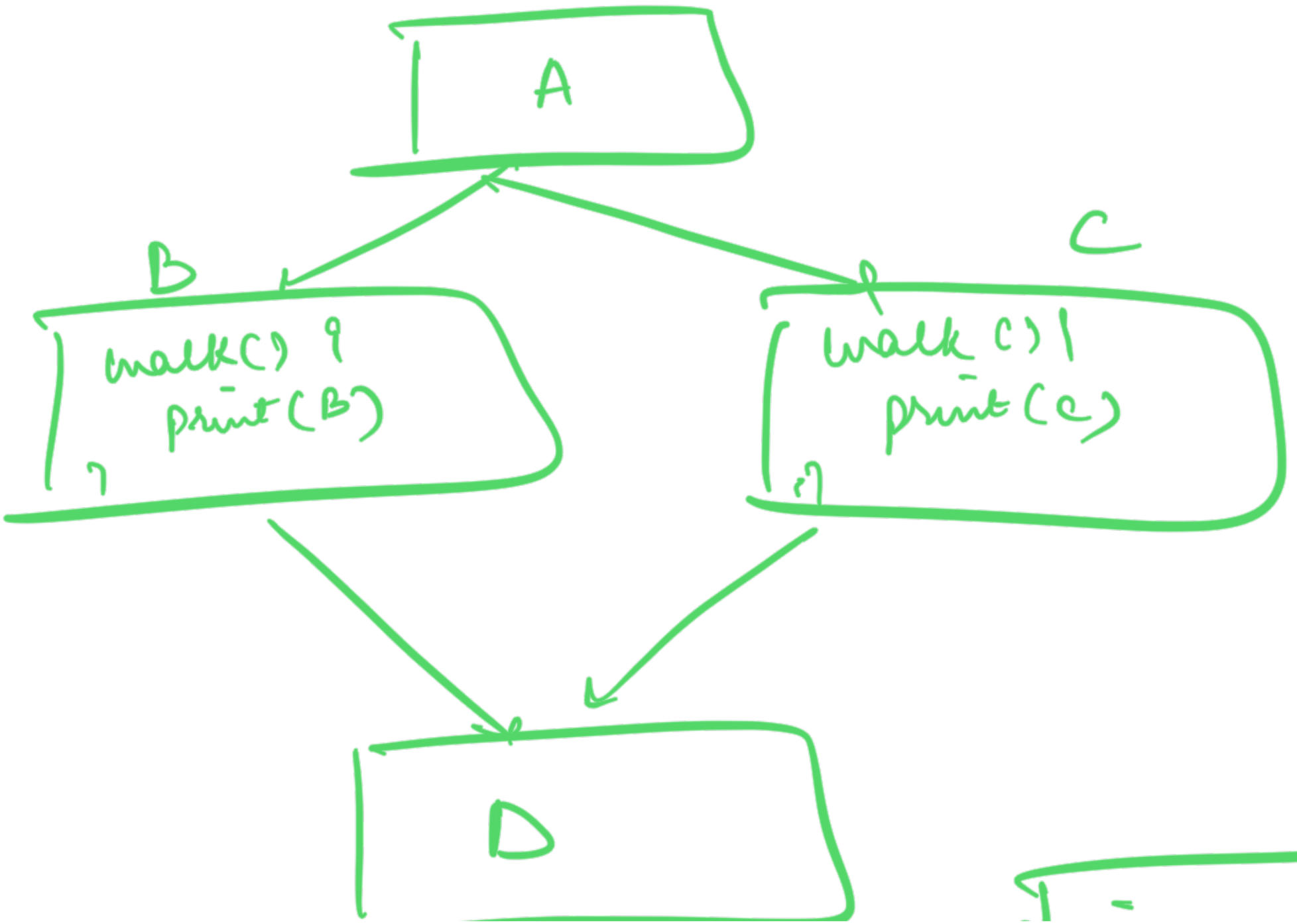
class Child extends Parent {

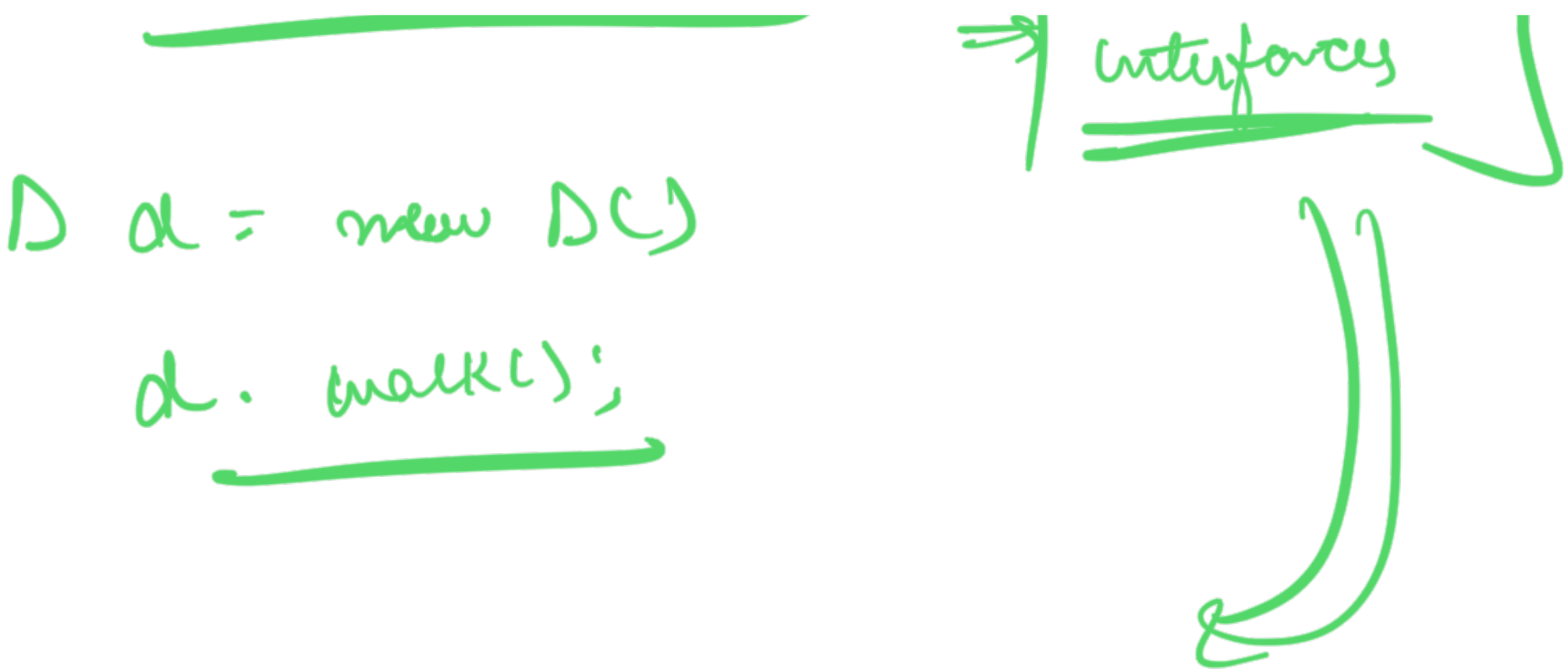
}

Java allows to extend ONLY ONE CLASS

→ No multiple inheritance

Diamond Problem






```
D d = new D();
d.walk();
```