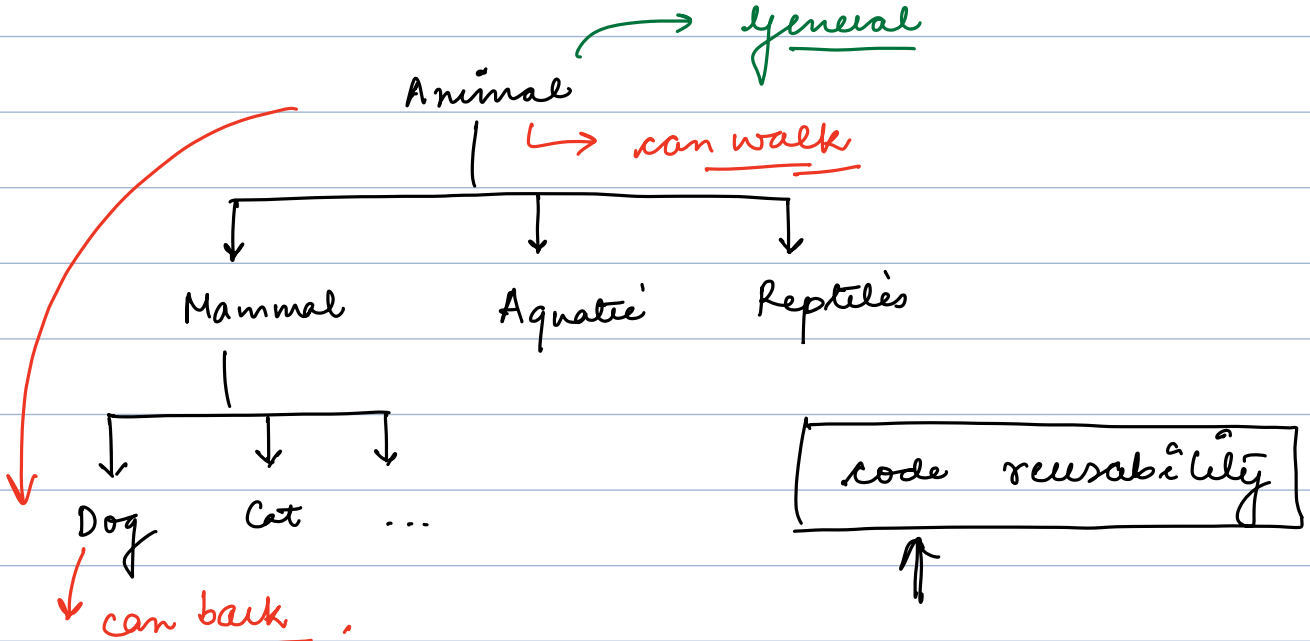
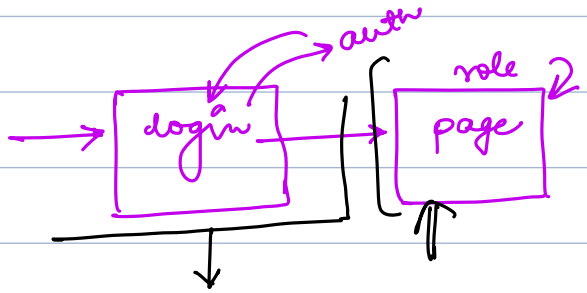
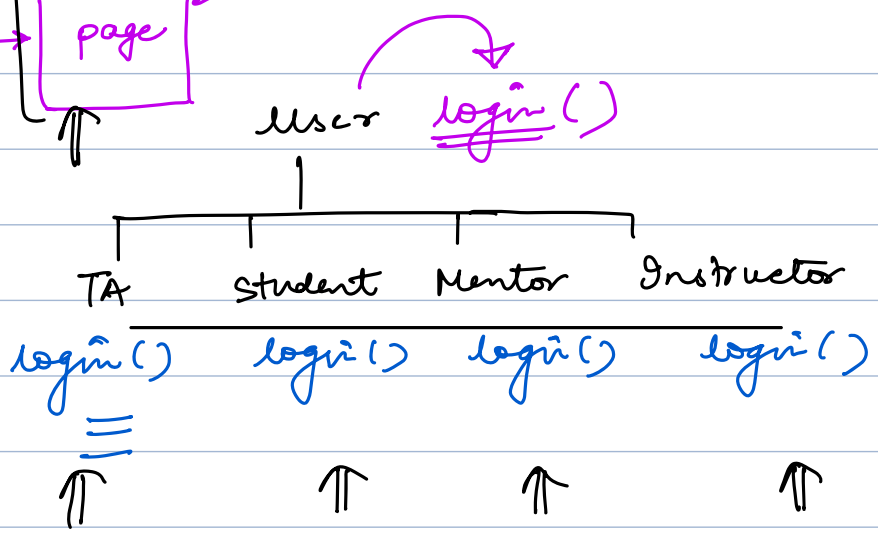
Inheritance and polymorphism

Inheritance

- Hierarchy







Parent
↑
child

inherits all the fields and methods of parent class.

+ private attributes are not inherited

A
↑
B

class A {

}

class B extends A {

}

class xyz.

private fun(int a) {

}

public fun(int a, int b) {

}

fun(5); ✗

fun(5,7); ✓

super → immediate parent

this → current.

B( ) {
⋮
}

B( ) {
this( );
}

C( ) {
super( );
}

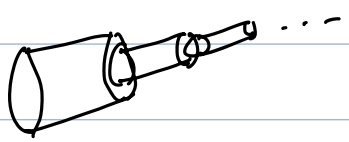
constructor chaining

- super() must be the first line
- this() must be the first line

- child class object creation will call parent class non-parameterized constructor by default.

Telescoping constructors

```
class B { int x, y, z;
    B() {
        ==
    }
    B(int x) {
        this();
        this.x = x;
    }
    B(int x, int y) {
        this(x);
        this.y = y;
    }
    B(int x, int y, int z) {
        this(x, y);
        this.z = z;
    }
}
```

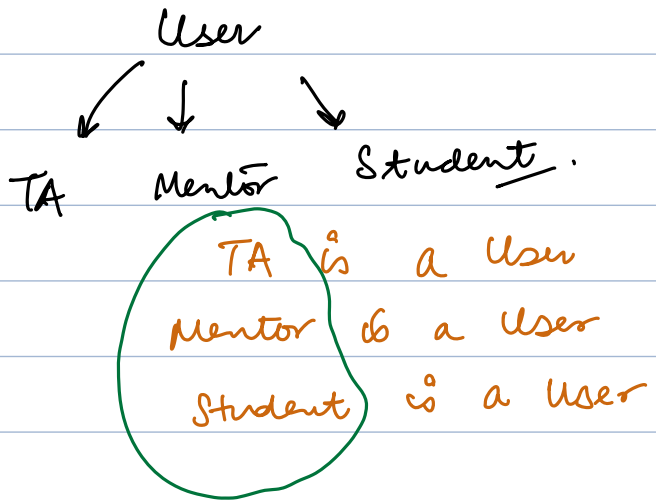


Break: 10:20 pm

Polymorphism → forms.

↓
many

Someone who has multiple forms.




```
print User names ( List<User> users) {

}
```

```
users.add(new Student());
```

```
User x = new Student();
```

```
Student y = new User();
```

Parent class ref can refer to child class object but vice-versa is not valid.

```
changepassword ( User x ) {
    ==========
}
```

changepassword (new Instr ( ));

extendAnimal

```
Animal ( ) {

    walk ( ) {
    ========
    }
}
```

```
Dog ( ) {

    bark ( ) {
    ========
    }
}
```

(I) Animal a = new Animal ( );

✓ a.walk ( ); a.bark ( ); ✗

(II) Dog d = new Dog ( );

✓ d.bark ( ); d.walk ( ); ✓

III

Animal a = new Dog();

✗ a.bark();

doSomet (Animal a) {

a.bark();

}