Day 8 - Access Specifiers, Polymorphism

20 July 2023 10:06

Access specifiers/modifiers

1) private
2) protected
3) default
4) public

1) private : -> only in the class

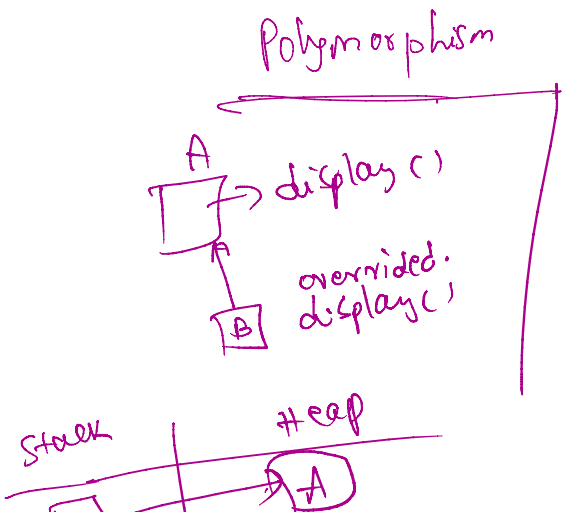
2) default : -> (No keyword)
-> accessible anywhere in the package.

3) protected : -> anywhere in same package
+
subclasses outside pacg.

4) public : -> accessible anywhere.

	default	private	protected	public
Same Class	Yes	Yes	Yes	Yes
Same package subclass	Yes	No	Yes	Yes
Same package non-subclass	Yes	No	Yes	Yes
Different package subclass	No	No	Yes	Yes
Different package non-subclass	No	No	No	Yes

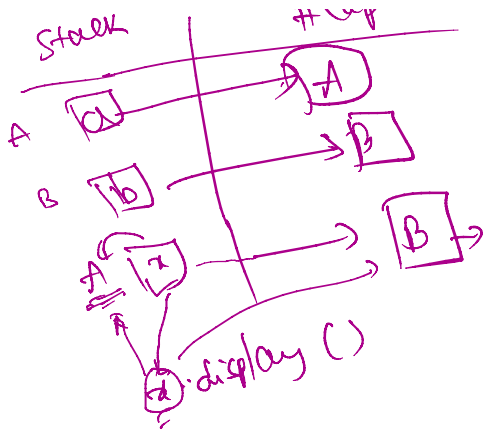
Polymorphism




```
A a = new A();
a.display()

B b = new B();
b.display()

A x = new B();
x.display()
```



x.display()

B b = new B()

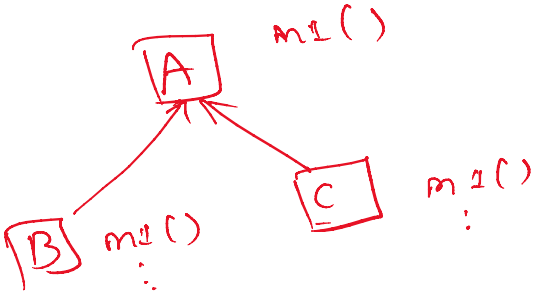
① b.display() ② b.another method()

A x = new B() ④

③ x.display()

x.another method() X NOT possible

NOTE: Parent ref. can access only overrided method in child class.

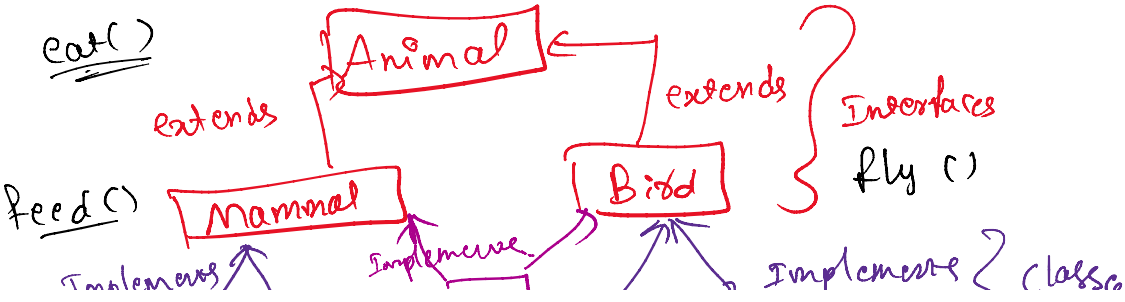


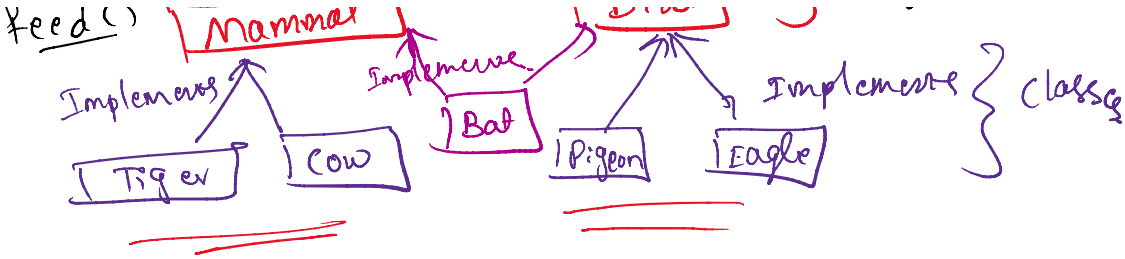
A x = new B()

x.m1()

A x = new C()

x.m1()





Try with Bird

```
System.out.println("\n---- Polymorphism Example ----");
Mammal mammal = new Cow();
mammal.eat();
mammal.feed();
System.out.println();

mammal = new Tiger();
mammal.eat();
mammal.feed();
```

Animal animal = new Tiger();

animal.eat() ✓

animal.feed() ✗

animal = new Pigeon()

animal.eat() ✓

animal.fly() ✗

Static

→ class level attributes.

class Demo {

public static void main(String[] arg)

{

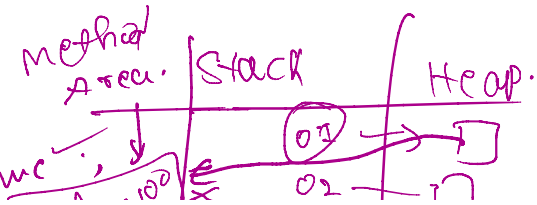
}

}

Demo demo = new Demo();

demo.main();

class Demo {



class Demo {

static int value;

static void m1() {

value = 100;

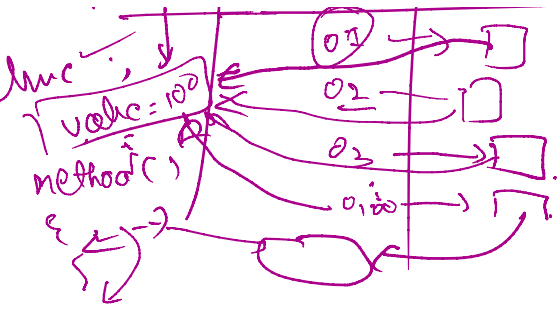
}

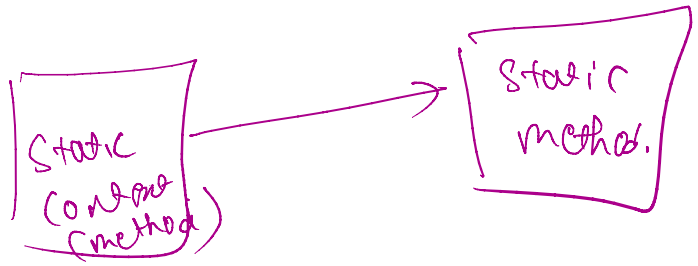
}

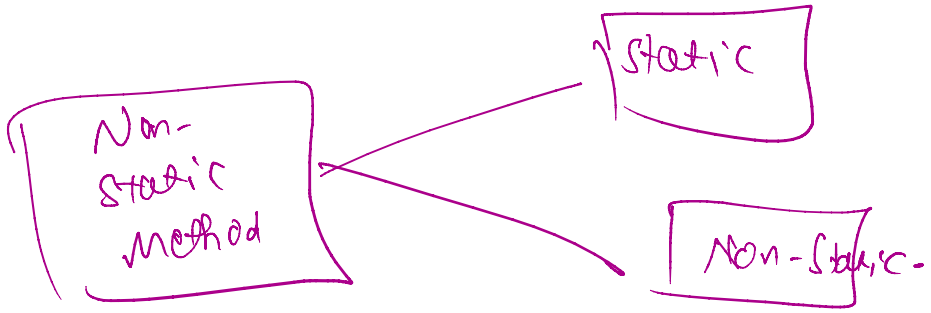
psvm ( ) {

Demo.value = 100;

}







* WAP to count how many objects of a class are created.

A a = new A()
b = new A()

→ count = 1
→ count = 2

class A {

static void display() {

class B extends A {

- - -

psvm void

class A {

public static void display()

}

}

public void display()

{

}

}

final → variable → can't be modified.
method → can't be overriden
class → can't be inherited.

final int value;

Const()

{

value = 100;

}

final void display() ← can't be overriden

{

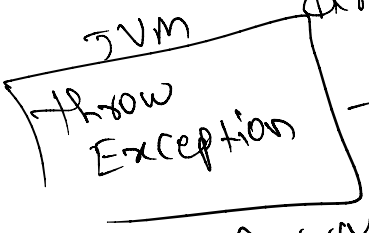
}

public final class A {

}

Exceptions

↳ Runtime errors.



Array[5] → 0, 1, 2, 3, 4

→ array[5] = 10;

1) ArrayIndexOutOfBoundsException
2) NullPointerException
3) ArithmeticException ← 5/0