LLD 2 OOP Terminology

→ Polymorphism

→ Interfaces

→ Coding Problems

Design Principles (SOLID DP)

→ SRP

→ Open Close Principle

Principle : Abstraction

Pillars: Encap, Inheritance

POLYMORPHISM ⇒ Having more than one form

⇒ Many forms

① Compile Time Polymorphism

② Runtime Polymorphism

Print Method has multiple forms

Sy of Polymo

⇓

Multiple form

print (String)

print (int)

fruit (String, int)

Method Overloading → A method

with same name but diff signature

↳ Eg of Polymorphism

① atleast the #OR Type of parameters is diff

int
float

print (String)

print (String)

int print (String)

int print (int)

int print (String)

float print (int)

How to decide of method Overloading

int a = p(1, 2, "hello")

p(int, int, str): float

p(int, str): int

float add(int a, int b) {

return a+b;

}

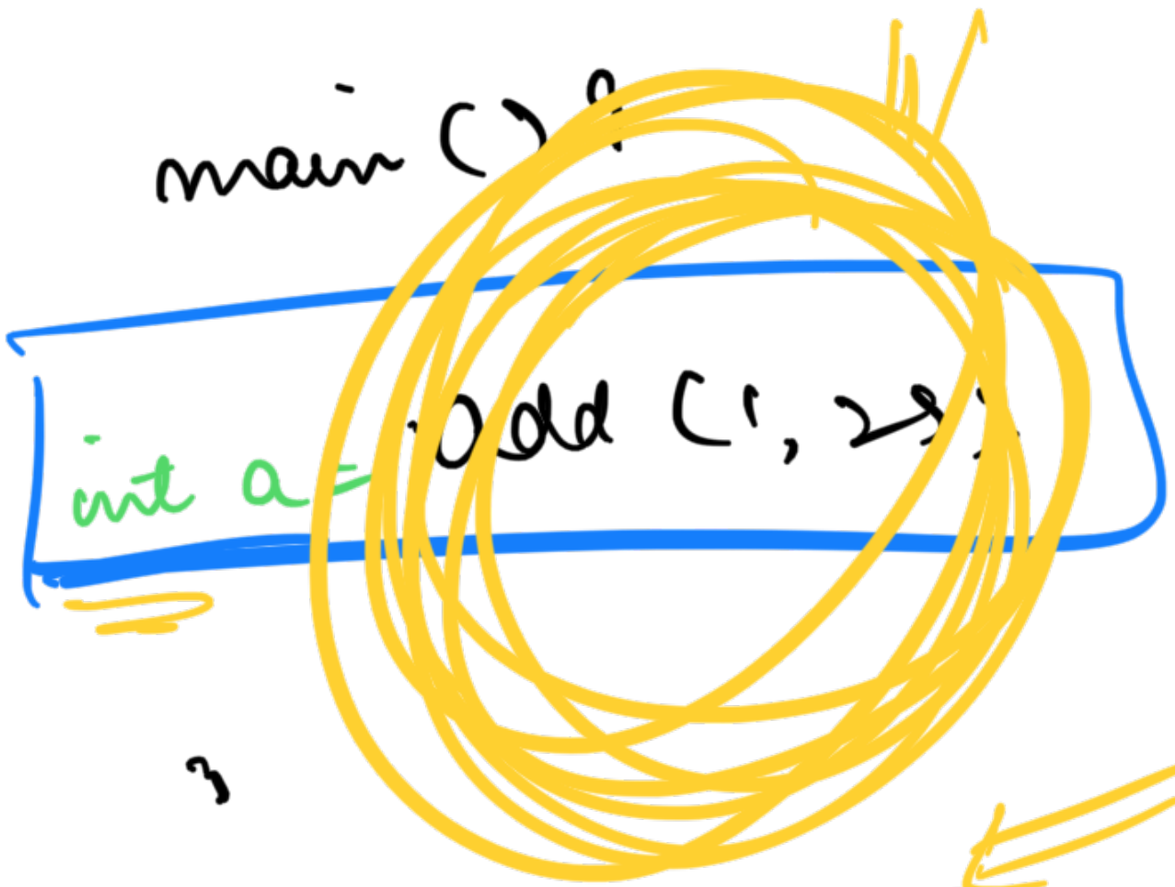
int add(int a, int b) {

return a+b;

main ( ) {

int a = odd (1, 2);

}



3

int a = 3

As associativity is right to left, there is no way for runtime to know what type of return value do I need.

int a = add (1, 2)

float c = String add (2, 3)

int add (1, 2)

Exact Type

list a = get list()

int print() ?

double print()?

cout (Yahoo)

cout (Google)

}

main ()

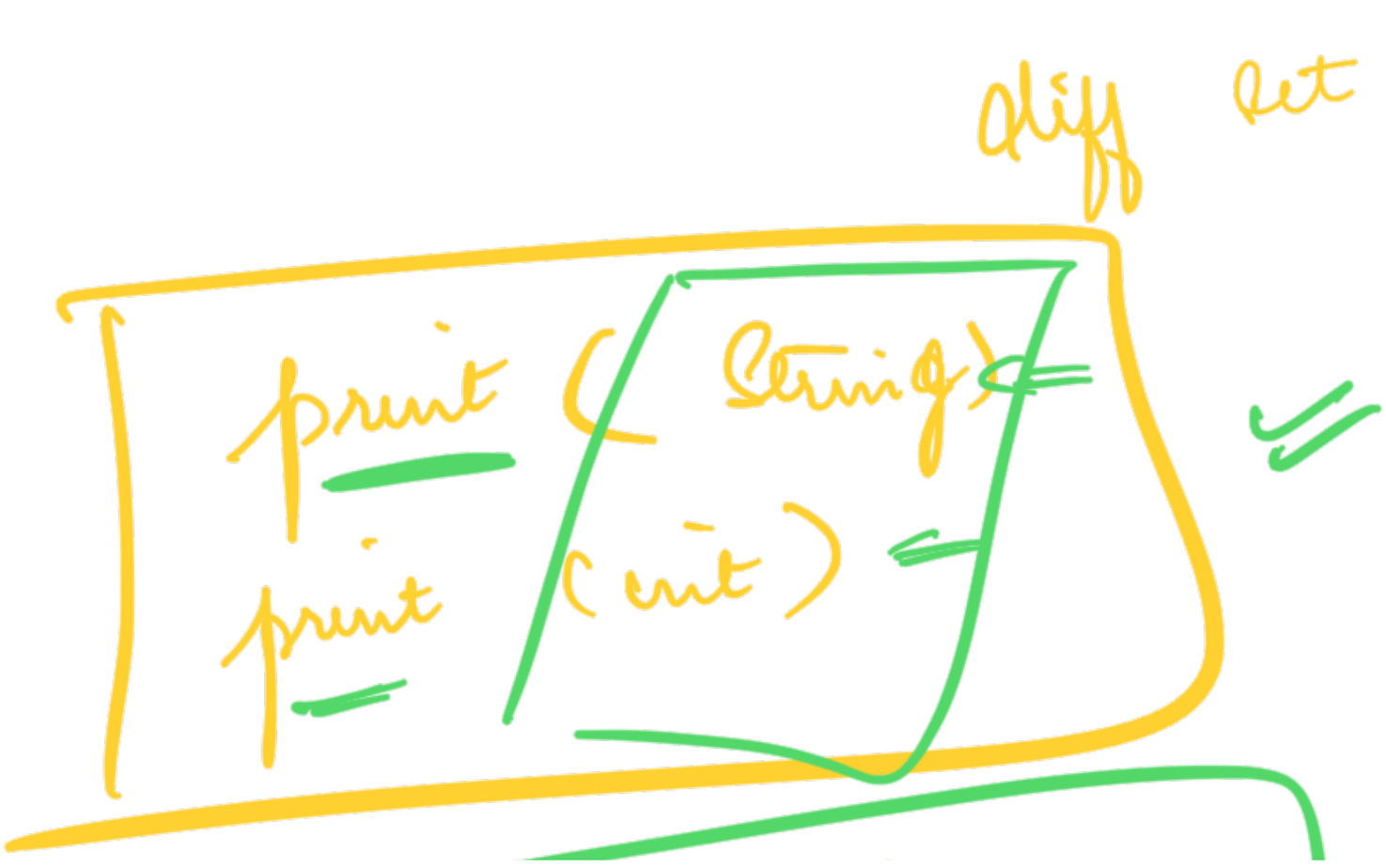
print ()

}

Polymorphism → Something having more than one form.

Many Form

Method Overloading : Multiple methods with Same name but diff set of parameters.



print (String)

print (String, int)

print (String, int)

print (int, String)

IF ONLY DIFF IS RETURN TYPE

→ COMPILE TIME ERROR

!= Method Overloading

Runtime Might

Not be able to
unambiguously
resolve the correct
f^n call.

Method Overriding → Supreme court overid the decision of high. court

↓

Changing / replacing

```
class A {

    doSomething() {
        print("I am A")
    }
}
```

```
class B extends A {

    doSomething() {
        print("I am B")
    }
}
```

}

To be overriden it should be EXACTLY SAME as Parent (Both Return Type + parameter)

B b = new BC();

b.doSomething()

Q^n When we were overriding, if number

type of parameter is diff.

Sequence

Not overriding

But Overloading

Inheritance

Animal

IS-A

Dog

Dog extends Animal?

?

int a = ____

Dog a = new Dog()

Dog

Dog

Animal a = new Dog()

Animal

Dog

Animal

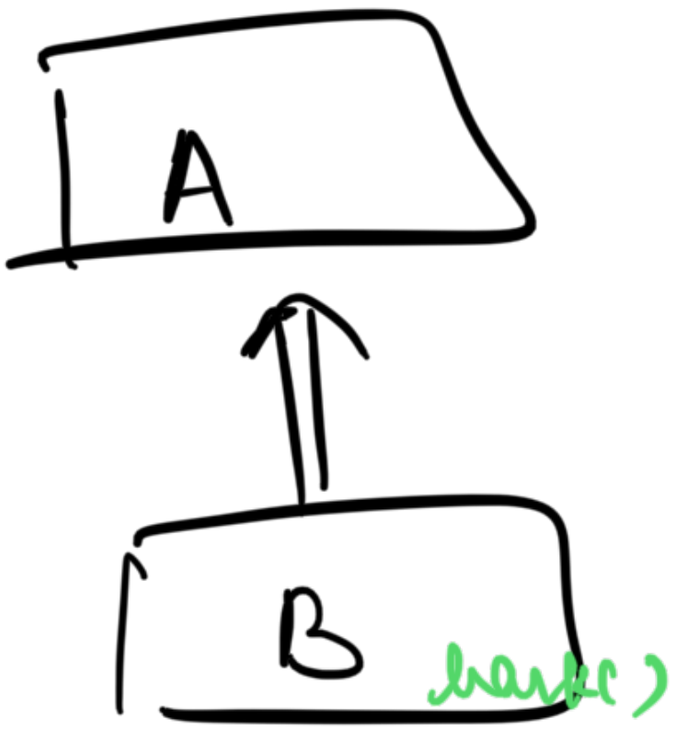
Dog d = new Animal();

even though dog IS-A animal

BUT Any animal MIGHT NOT Be a Dog.

Dog d = (Dog) new Animal();

d.bark()



B b = new B();

A a = new A();

B b = new A();

A a = new B() ✓

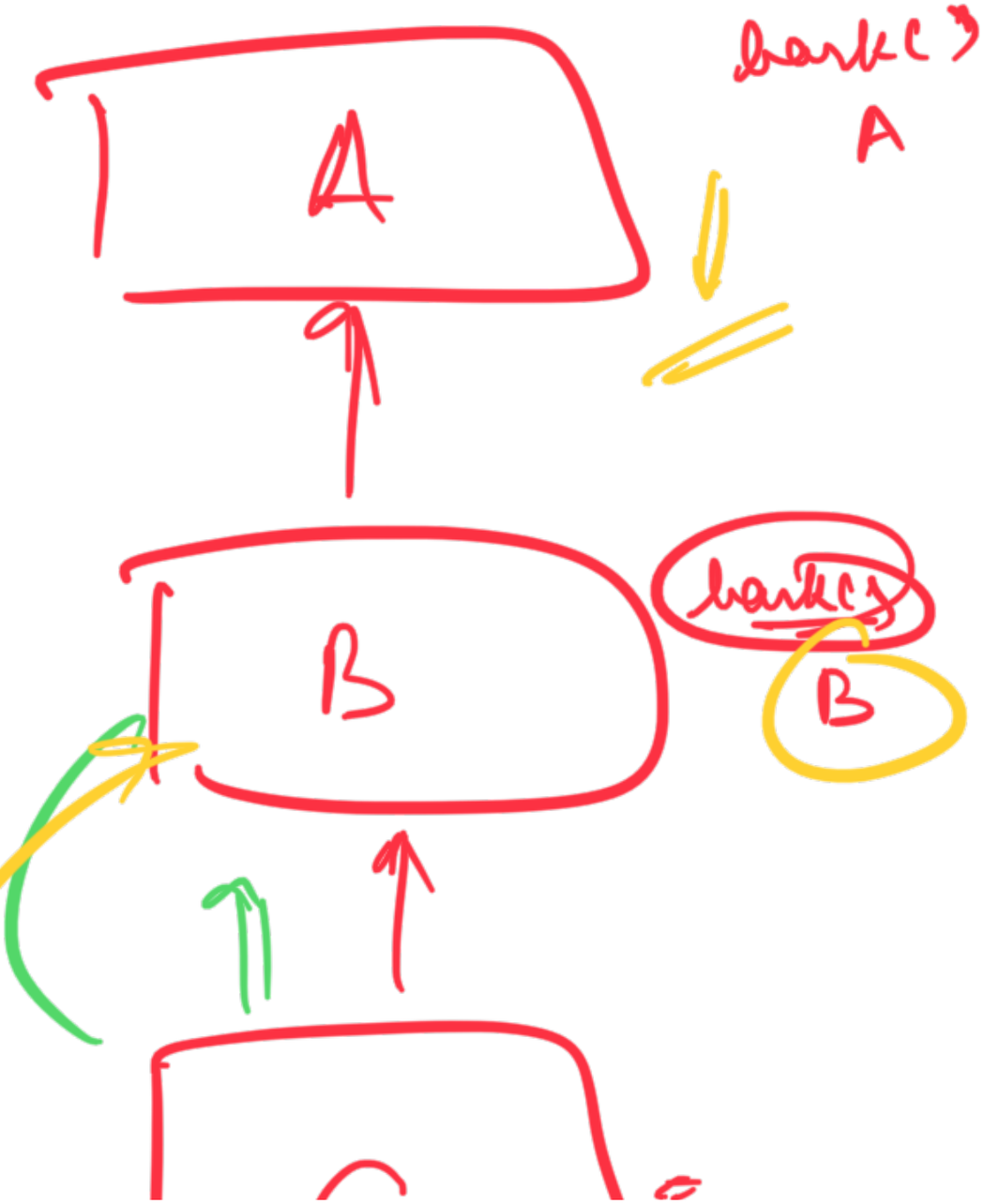
```
1 ⇒ A b = (A) . new A()

2     b . bark()
```

a.do (); }

A a = new B();

Bog.

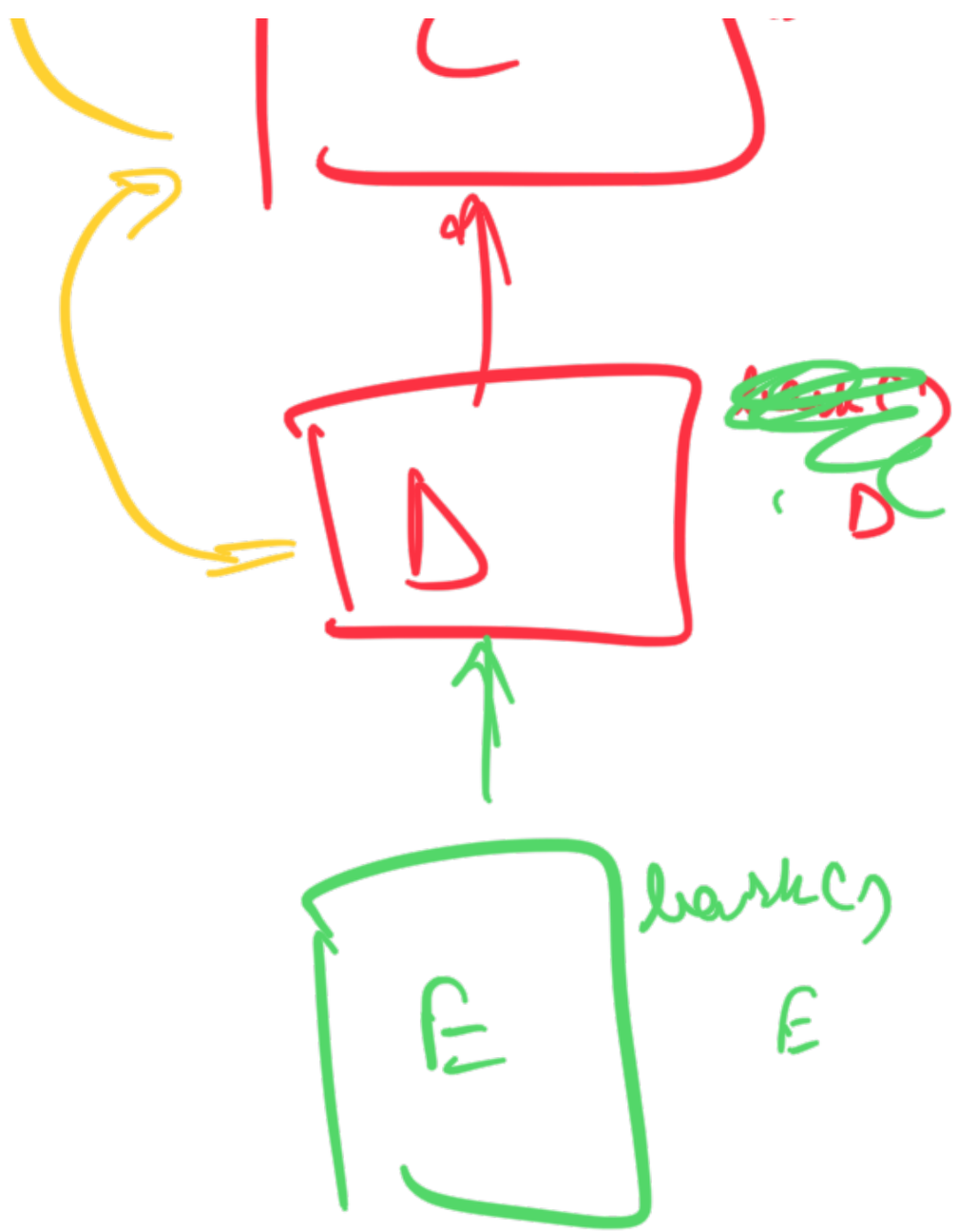


bark()
A

bark()
B

A a = new A();
a.bark();

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




```
C c = new C();
c.bark();
```


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

Polymorphism is of 2 type

1. Compile time polymorphism

① Compile time Polymorphism

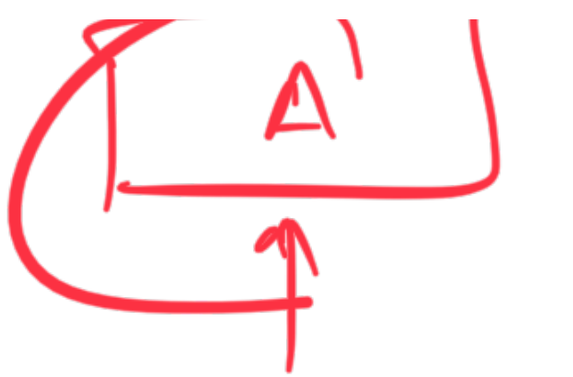
When you know which code will exactly run at compile time stuff.

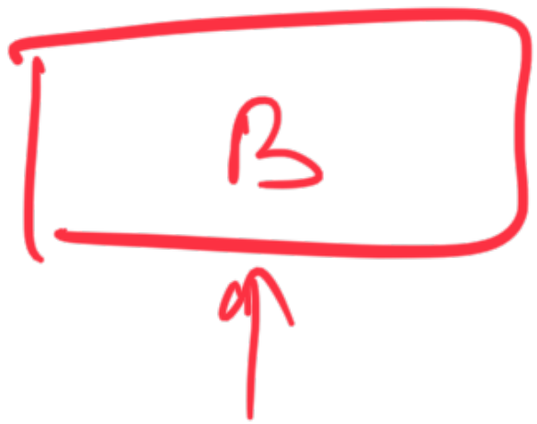
→ Method Overloading

② Runtime Polymorphism

Exact code that will run will only be known at run time.

→ Method Overriding





max^m Level Of Abstraction

having a broader idea

being more general