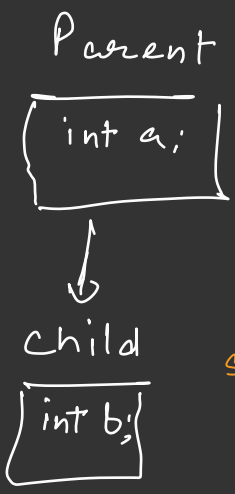
Pillars of OOPs

1. Inheritance
2. Polymorphism
3. Encapsulation
4. Abstraction

1 Inheritance

↳ Same as inheriting properties from parent to child



child class can inherit the properties of parent

Syntax

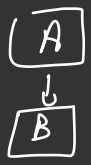
```
class Parent {
    int a;
    void printA() {
        syso(a);
    }
}
```

```
class Child extends Parent{
    int b;
    void printB {
        syso(b);
    }
}
```

```
Child c = new Child();
c.b
c.printB();
c.a
c.printA();
```

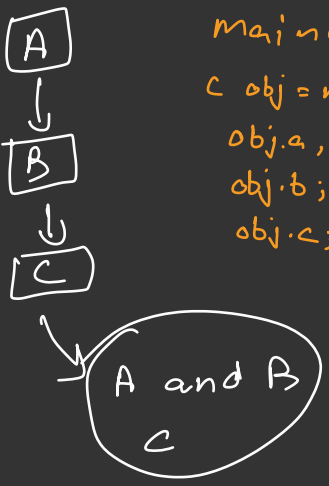
Types of Inheritance

(1) Single Level ⇒



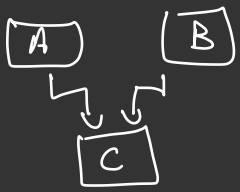
(2) Multilevel ⇒

```
class A {
    int a;
}

class B extends A {
  int b;
}

class C extends B {
  int c;
}
```



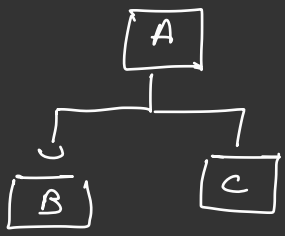

```
main()
C obj = new CC);
obj.a,
obj.b;
obj.c;
```

(3) Multiple Inheritance



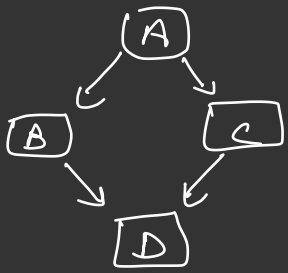
Not allowed in java classes

④ Hierarchy Inheritance



⑤ Hybrid Inheritance

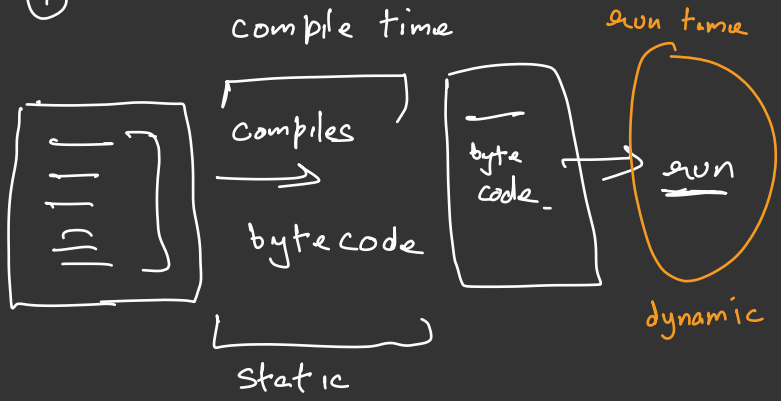
not allowed



Polymorphism

many — form

①



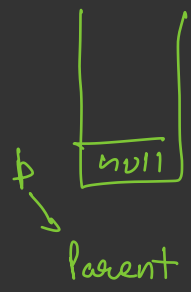
1. Compile time / Static
2. Run time / Dynamic

Compile time / static

function overloading

```
void sum (int a, int b) {
    syso (a+b)
}

void sum (int a, int b, int c) {
    syso (a+b+c);
}

void sum (double a, double b) {
    syso (a+b);
}

main() {
    sum(2, 3)
    sum(2, 3, 4)
    sum(2.0, 3.0)
}
```

Parent p = new Parent()



Parent a = new Child();

Run time Polymorphism / Dynamic

Final key word

final void area() {
}

→ Prevents Overriding

final class Shape {
}

→ Prevent inheritance