Inheritance

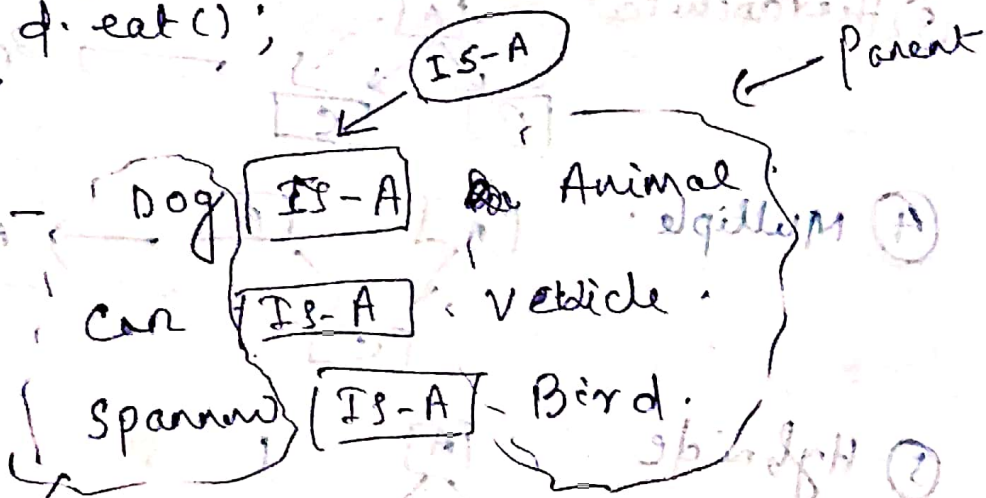
→ Inheriting the properties of parent class into child class.

OR → It is the procedure by which one object acquires all the properties & behaviours of a parent ~~class~~ object.

```
class Animal ──────────→ Parent / Super class
{
    void eat()
    {
        sop("I am eating");
    }
}

class Dog extends Animal ──→ child class / sub class
{
    psvm()
    {
        Dog d = new Dog();
        d.eat();
    }
}
```

Relationship:-



→ Inheritance [IS-A] Relationship.

*) Acheive code Reusability.

*) We can achive polymorphism.
(i.e. method overriding).

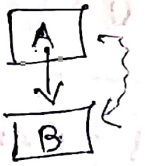
disadvantages

*) If ~~son~~ one change occurs in parent class that will affect to all the child class
(i.e. classes are tightly coupled)

Types of Inheritance.

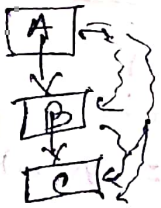
Java

1) Single



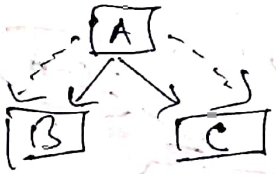
→ supports

2) Multilevel



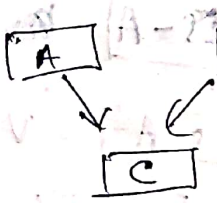
→ supports

3) Hierarchical



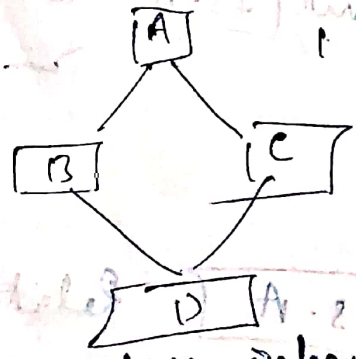
→ supports.

4) Multiple.



→ Not supp

5) Hybride

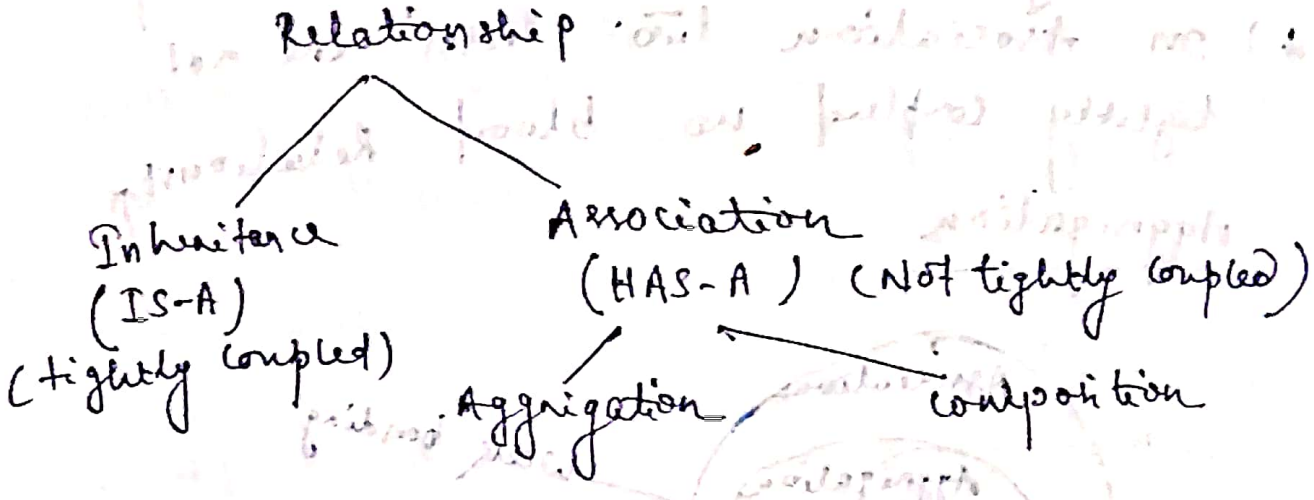


→ Not supp

*) when a class inherits the property of another class then all the properties of parent class are not inherited. e.g. constructor of parent class, private method

Relation ship between classes

*) what are the Relationship between classes.



Advantages

1) code Reusability
2) cost cutting.
3) Reduce Redundency.

(HAS-A)

```
class Student
{ String name;
  int rollno;
}
```

Association

Student HAS-A name

Student HAS-A rollno.

eg:

```
class Engine
{
}
class Car
{
 Engine e=new Engine()
}
```

IS-A

Car extends Engine

Using new keyword. The Relationship builds.

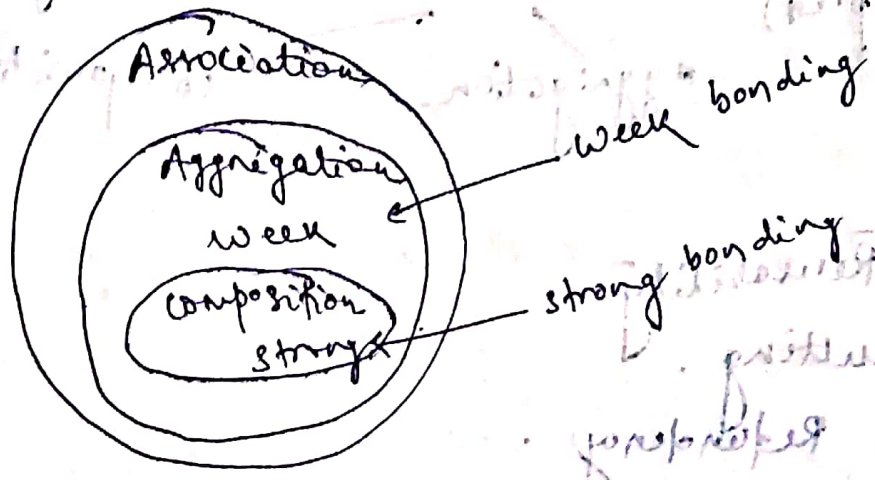
Car HAS-A Engine.

HAS-A

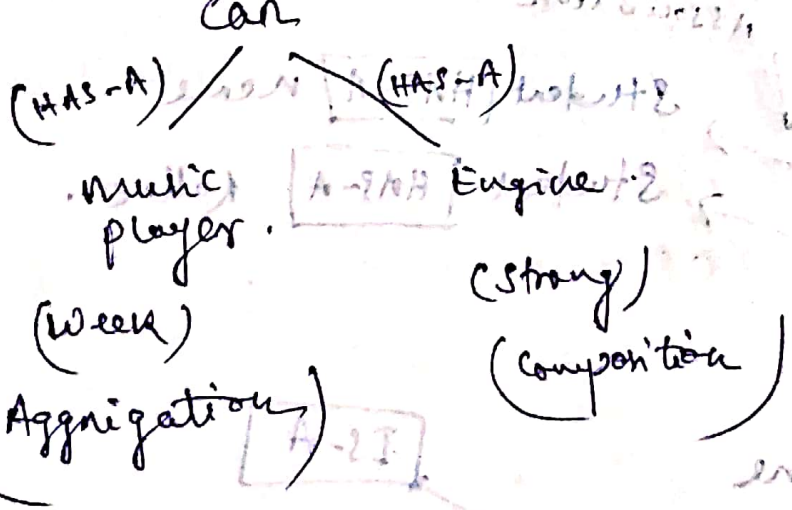
*) In Inheritance two classes are tightly coupled. i.e. blood Relationship

*) In Association two classes are not tightly coupled no blood relationship

Aggrigation



e.g.



Relationship between classes.

Inheritance.
(IS-A)
→ extends
– blood Relationship
– tightly coupled

Association.
(HAS-A)
→ Ref variable, new
– non blood Relationship
– not tightly coupled / losely coupled.

Aggregation
– Weak

Composition
– Strong

e.g.
Vehicle
IS-A
Car
HAS-A
Mplayer
HAS-A
Engine

Super keyword:

1*) super is used to refer immediate parent class instance variable

2*) used to invoke immediate parent class method

3*) used to invoke immediate parent class constructor.

e-g1)

```
class A
{
    int i=10;
}
class B extends A
{
    int i=20;
    void show(int i)
    {
        S.O.P(i); // 30
        S.O.P(this.i); // 20
        S.O.P(super.i); // 10
    }
P.S.V.M( )
{
    B ob = new B();
    ob.show(30);
}
}
```

immediate parent class method

e.g. 2

```
class A
{   void m1()
    {   s.o.p("in A");
    }
}
class B extends A
{   void show()
    {   super.m1();
    }
}
P.S.V.M()
{   B ob = new B();
    ob.show();
}
```

immediate parent class Constructor.

e.g.

```
class A
{   A()
    {   s.o.p("I am in A");
    }
}
class B
{   B()
    {   s.o.p("I am in B");
    }
    P.S.V.M(String[] args)
    {   new B();
    }
}
```

Output:

I am in A
I am in B