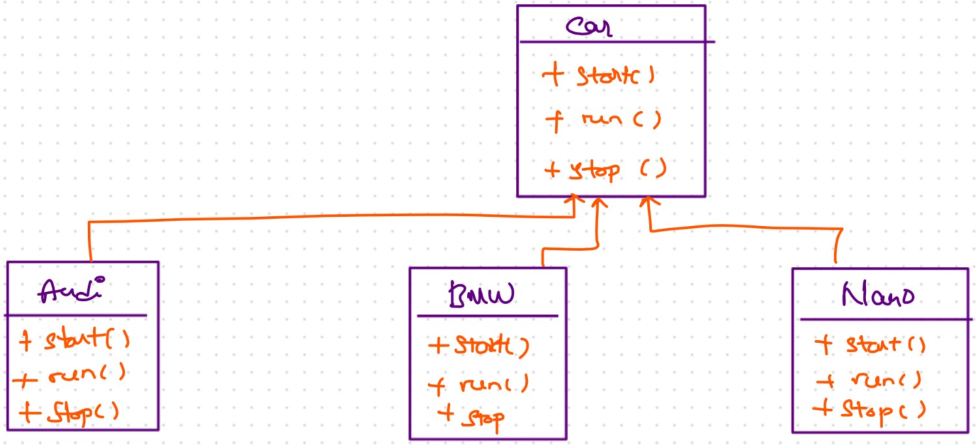
Design Patterns

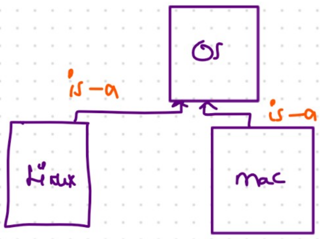
It is just a way of writing code

Factory Design Pattern:

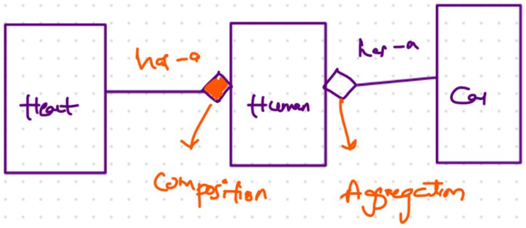


→ refer program FactoryDesignPattern.java in Day21

① is-a-relation



② has-a-relation

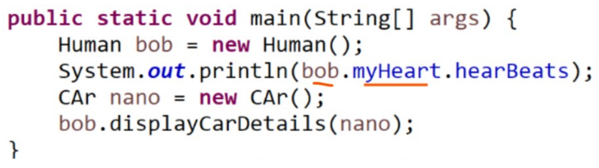


These are of 2 types

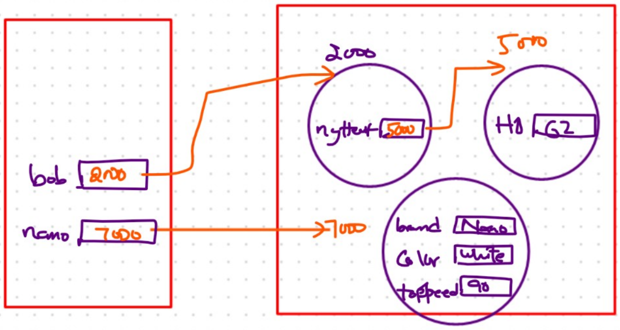
(1) composition

(2) Aggregation

memory Mapping



```
public static void main(String[] args) {
    Human bob = new Human();
    System.out.println(bob.myHeart.hearBeats);
    CAr nano = new CAr();
    bob.displayCarDetails(nano);
}
```



Note:

→ In a real world, we not only find is-a relation among the object but also has-a relation

→ Parent - Child relation is considered as "is - a" relationship

→ "Has - a" relation is achieved in 2 ways

(1) Composition

(2) Aggregation

Composition: When the main object is destroyed, then The subobjects also gets destroyed, then such objects are considered as composite objects

→ Sub objects are created when the main objects created

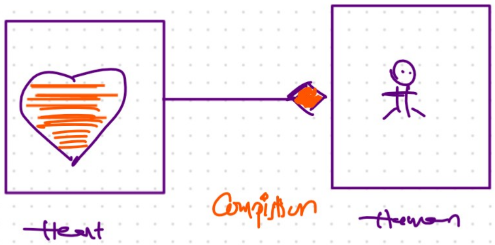
→ We can achieve this by making sub-objects as The instance variables in The main object & initialize them in constructor call

example:

(1) Human has Heart

(2) Car has Engine

(3) Computer has OS



Aggregation: If there is "has - a" relation b/w two objects & if one object is destroyed, if the other object remains to exist, then such objects are considered as aggregate objects

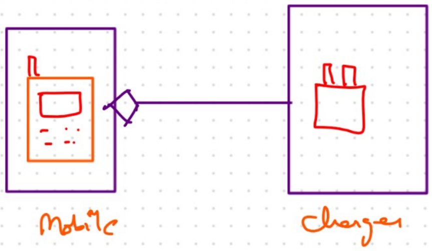
→ In this case, the objects are created seperately & they are not dependent for creation, deletion of another object

→ we can achieve This by passing the object as parameter for a method of another class

example

(1) Human - car

(2) Mobile - Charger


Mobile
Charger