* Object ⇒

Object is anything which hav

i) State
ii) Behaviour
iii) Identity
iv) Responsibility

① State ⇒

Giving value to Attribute

Attribute + value = State

Example ① →

Attribute	Value
Name	Board
Type	Wooden
Colour	white
Shape	Circle
Size	7m / Radius

Example ② →

with-out Attribute we cannot specifies state.

value:
6 feet
Sunil
Brown
Pune

{ from this example we can predict it can be person or shop.... so Attribute is Important

Example ③ →

Watch

Attribute	value
Name	Fireboltt
Price	2500/-
Colour	while grey
Type	digital

Example ④ →

Smartphone

Attribute	value
Name	IQOO
price	25000/-
colour	Black
processor	Mediatech 7200
Camera	50 Mega Pixel rear

Example ⑤ →

Pen

Attribute	value
Name	Trimax
Price	50 Rs
Type	Gel pen
colour	Blue

③ Behaviour ⇒

How an object response to outside world (Response giving)

Example →

(i) Fan :→

When we turn on the switch fan rotates while when we turn off the switch fan stops rotating.

(ii) Pen :→

Used for writing on papers coloured pen used for highlighting the main points.

③ Identity ⇒

Uniqueness associated with every object.

Example :→ i) Phone no.
ii) Adhar no.
iii) Roll no.

When we need Identity ?.

→ To differentiate difference bet^n two object having same state & behaviour

* State & Behaviour are Inseperable in Real life

④ Responsibility ⇒

Role of an object in particular program.

Example →

i) Responsibility of teacher to teach

ii) Responsibility of Pen to work while writing

iii) Responsibility of T.V to show movies, shows etc.

iv) Responsibility of fan to gave us accurate wind in appropriate speed.

Examples ⟹

① Earbuds →

a) State

Attribute	value
Name	Wings phantom
Type	Gaming
Colour	White
latency	50 MS

b) Behaviour

When we turn on earbuds it gave us sound in each bud i.e left, right when pressing on right bud it unable Game mode.

c) Identity

It has specific Identity no. on box to get is unique Earbuds.

d) Responsibility

Its responsibility to gave us nice bass, low latency and accurate voice response while calling

* No program became Object Oriented implicitly

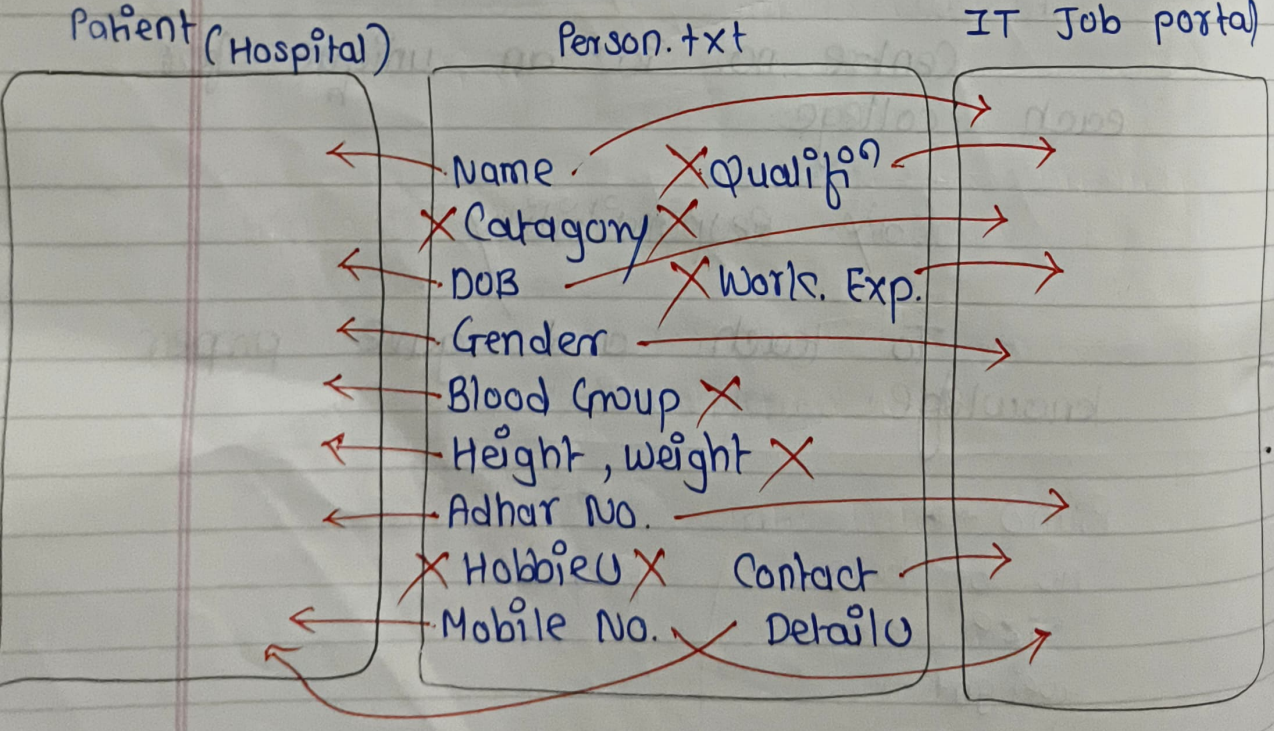
* We need to follow the steps to make a program Object-Oriented

① Abstraction
② Encapsulation
③ Inheritance
④ Polymorphism

① Abstraction =>

Abstraction is a process of "selective ignorance"

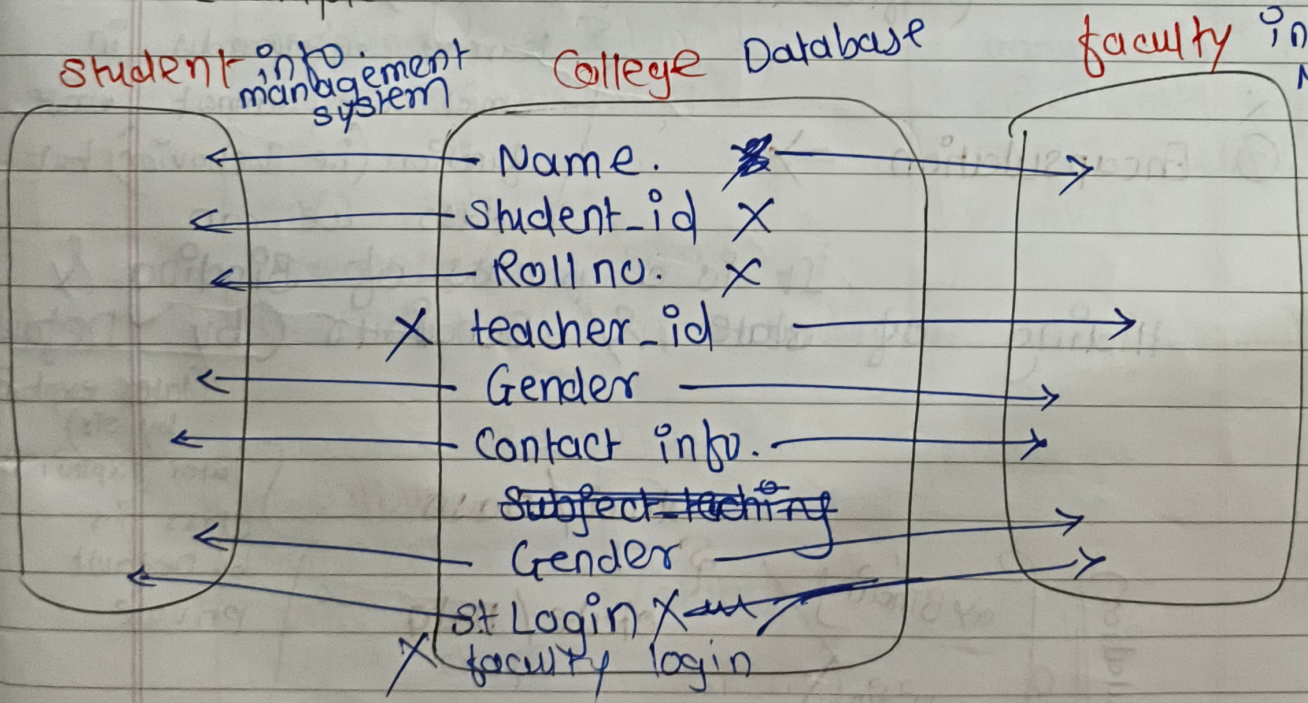
Example →



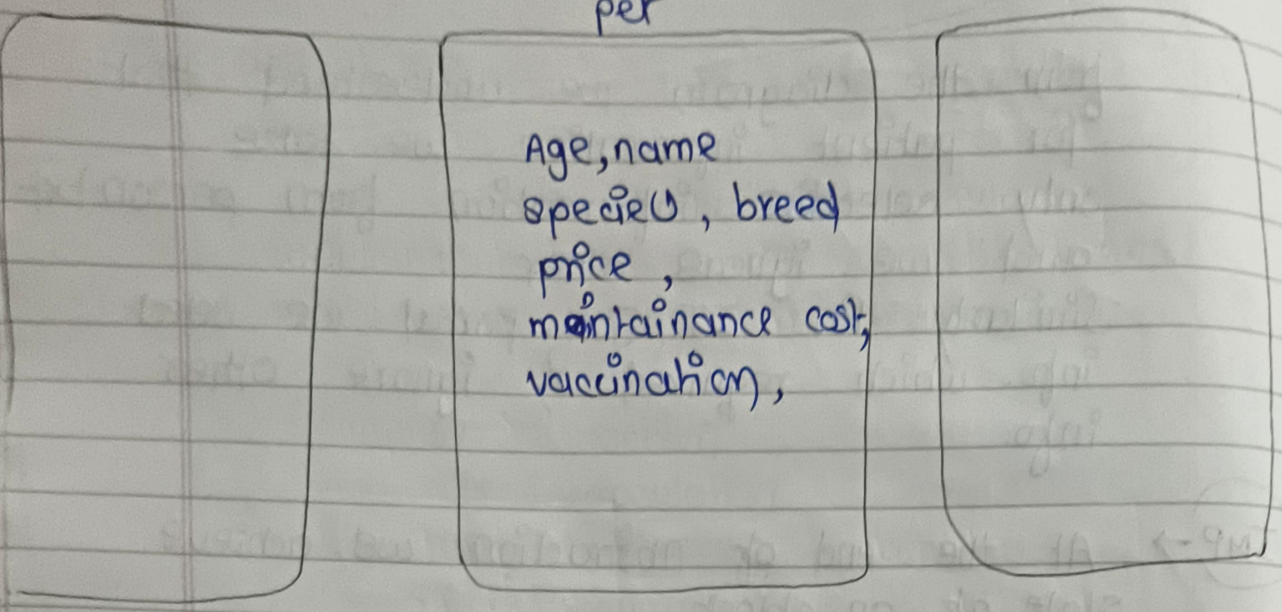
from the diagram we understand that for patient information we take only selective information from person.txt and we ignore other
Similarly for IT Job portal we select info. which required & ignore other info.

IMP → At the end of Abstraction we achieve state of an object
(i.e we get person information in patient box but we didn't know him i.e object but we achieve state)

Example →



Hospital



* Encapsulation is completly implemented by keyword class
* keyword struct fail to implement Encapsula°

(Afterwards learning)

learned afterwards in struct we cannot write function (i.e behavior) but in class we can

② Encapsulation ⇒ It is a process of Binding & Hiding of state & Behaviour (by Default)

(Later explain by sir)

later explained class is by Default private

Encapsulation { a) Binding, b) Hiding } of { struct / class { state / Data, Behaviour / function } }

Class is way of Binding & Hiding state & Behaviour

* Why to bind ?.

→ State & Behaviour are inseparable in real life so in programming also it is inseparable

* Why to hide ?.

→ In real life data is hidden (suppose a person name (Ankit) we didn't know his attributes & values Hence it is Hidden)

Example ⇒

① Fan

no.of blades : int company_name : char[] price : int
on() , off() increaseSpeed() decreaseSpeed()

(state)

(behaviour)

② Pen

name : char[] price : int colour : char[]
towrite()

③ Earbud

name : char[] price : int type : char[]
Music() Calling() Gaming()

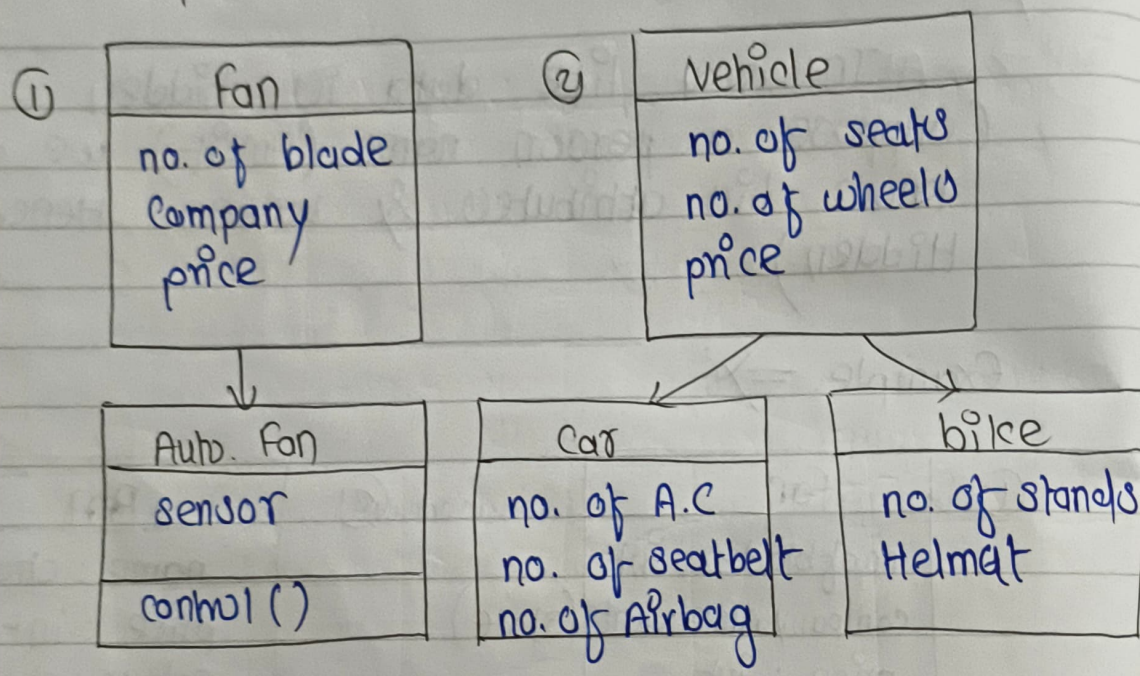
(state)

(behaviour)

③ Inheritance ⇒ It is way of use already defined state & behaviour with is a relationship.

It is a way of reusing predefined class, Already defined class with is a relationship.

Example →

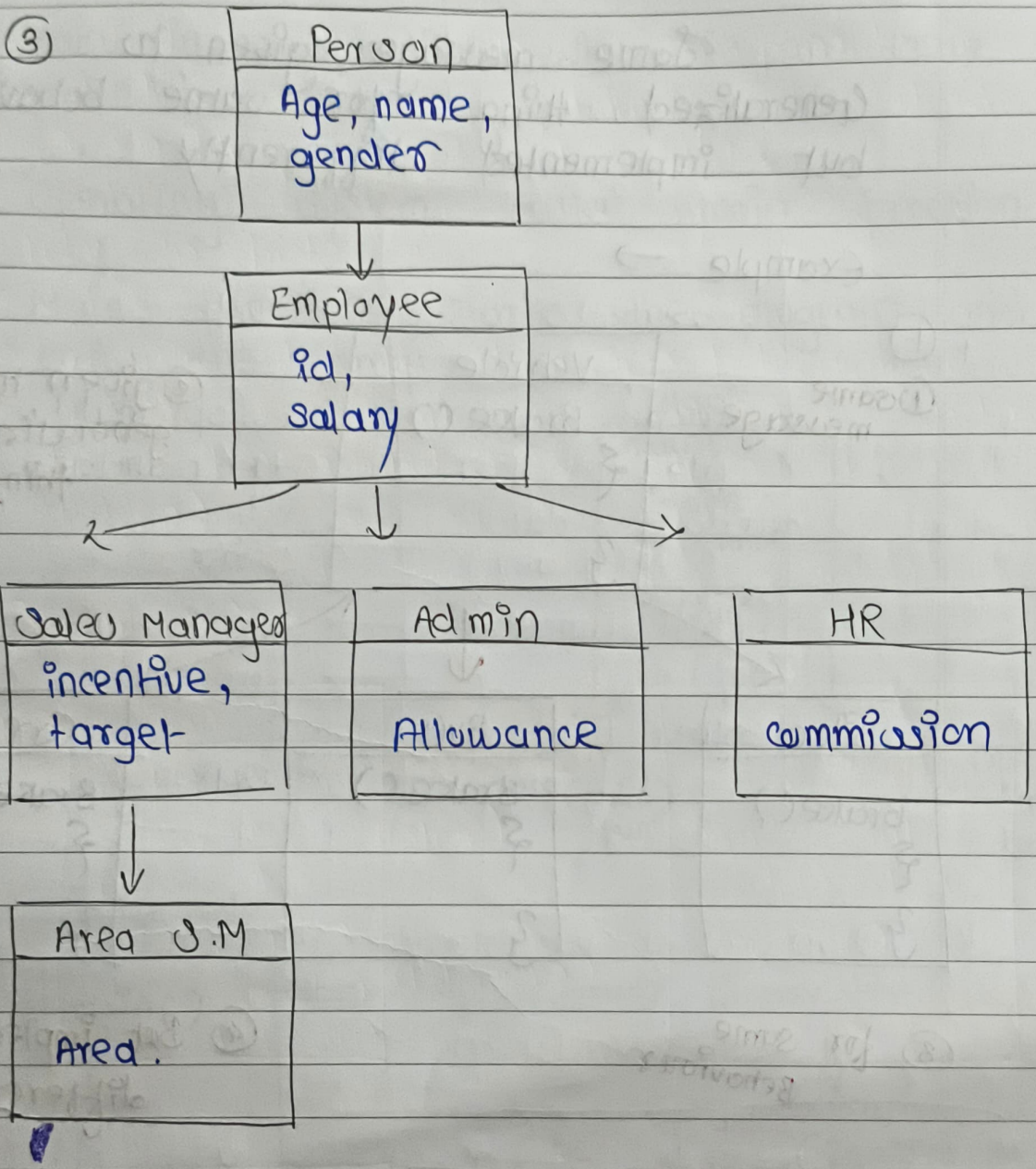


* Upwards to inheritance heirarchy we get generalization (General) common

* Downwards to inheritance heirarchy we get specialization. (Special) unique.

Example →

③



* Sequence of object orientation 4 pillar is important
i.e if we had to get inheritance of particular object we wanted to get first its encapsulation & before it Abstraction

④ Polymorphism ⇒

Same message given to Generalized thing for same behaviour but implemented differently

Example →