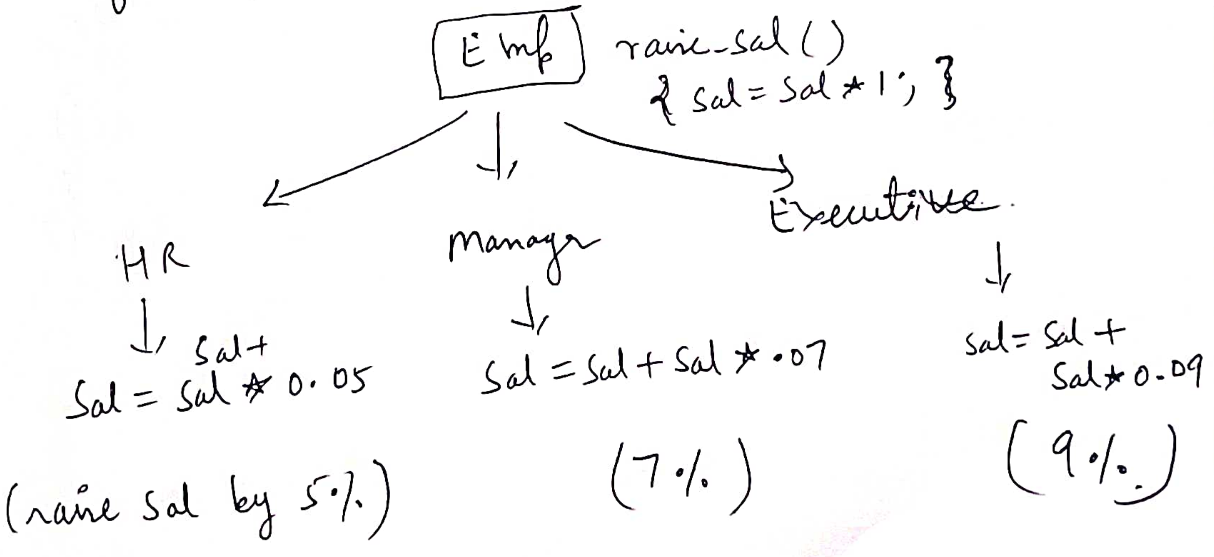
Implementing a Virtual Fun Example

(Unit - 4)

- 3 steps must be followed to implement a VF :

1) Create a VF in base class.
2) override / redefine this fun in derived classes.
3) Call this fun for derived class obj using base class ptr/ref..

EX. Create emp class
- It has 1 var called sal;
- It has a fun called raise_sal()
wh. shud the sal same i.e sal = sal * 1;
- Create derived classes HR, mgr, exe.
- Each class will override the raise_sal()
fun & use diff criteria to raise sal.



• Inside main :
  • Create obj of each class.
  • Create a base class ~~ptr~~ ref.

1) Point this ref to HR_obj & call raise_sal()
2) - - - - - - - Mgr_obj & - - - - -
3) - - - - - - - - - - exe_obj - - - - - - - - -.

Since this fun is declared as virtual, so each obj's own raise_sal() fun will be called in all the 3 cases.

Ans) See code :

- Summary
- Imp Points

Q) What is a VF?

A) A VF allows a base Clan ptr/ref to call a derived clan fun.

- In case of non VF, a base Clan ptr/ref can ~~ot~~ only call a base Clan fun; it can't call a der Clan fun.

Q) How to implement a VF?

A 1) Create a VF in base class
2) O/ride in der Clan
3) Use base ptr/ref to call.

Q) What is the correct way to use a VF?

A) VF, O/riding, Base ptr/ref must be used jointly to implement a VF

- This combination provides runtime Polymorphism.

Q) Characteristics of VF?

A) 1) Can't be static / friend

2) shud be declared as vir

3) Vir ~~[illegible]~~ constructor can't be created, but vir ~~cons~~ destructor can be " .

4) VF shud be overridden in der class.

5) " " " called using base ptr/ref.