DP

* where you will use it ??
* interview based ques.

LLD-2

2 lectures

(1 → 2, 0.5 → 1.5)

SOLID

8-10 lectures ⇒ DP

- Structural
- Creational
- Behavioural

DP :- Chakravy

Structure / formation pattern

Design patterns ⇒ specific structure of code

24 DPs ⇒ Gangs of four

introduced

10-11 DPs ⇒ ~90% of all cases

⇒ SOLID

S → Single Responsibility principle
O → Open closed principle
L → Liskov substitution principle
I → Interface segregation principle
D → dependency inversion principle

* principles ≠ rules

* try to make your code as modular as possible

SOLID

engineer ⇒ 9 hours ⇒ 1 hour

[8 hour]

12% ⇒ engineer spends time writing code

→ meetings :- product req., Code reviews, estimations, task breakdown, design disc.

→ bug fixing

→ PR review

→ documentation

extensionability ≠ scalability

→ testing

→ Chai | coffee | gossip etc.

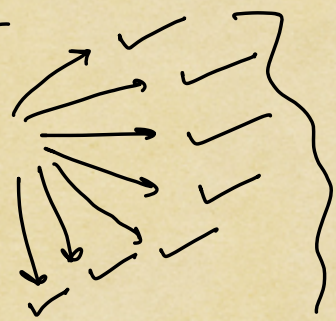
majority → reading, maintainence, extensibility, testing

- read
- maintain
- extend
- test

maintain, extend ⇒ keep the code running as it is
ability to add more add

SOLID ⇒ helps you to make code more (REMT)

log4j ⇒ V1.1.1



⇒ Crowdstrike bootloop MS issue

core module
↓
basic | fundamental dependencies
↓
version at 1 place

⇒ SRP

Single responsibility principle

any block of code should exist to handle a single responsibility

- class
- method
- block

class Car {

String cartype;

startEngine for multiple types

void startEngine() {

if(cartype == ev) {

} else if(cartype == hybrid) {

}

}

ev

cartype = ev

petrol

cartype = petrol

hybrid

cartype : hybrid

⇒ multiple if/else

⇒ readable (X)

⇒ maintainable (X)

⇒ extend (X)

⇒ test (X)

⇒ OPEN - CLOSED PRINCIPLE :-

Open for extension

close for modification

⇒ block of code [class, method, blocks] is a serving a purpose,

not allowed (should not) modify it.

⇒ add more code (✓) ⇒ allowed to extend

⇒ modify existing code (X) ⇒ not allowed to modify

SOLID

S R P

O C P.