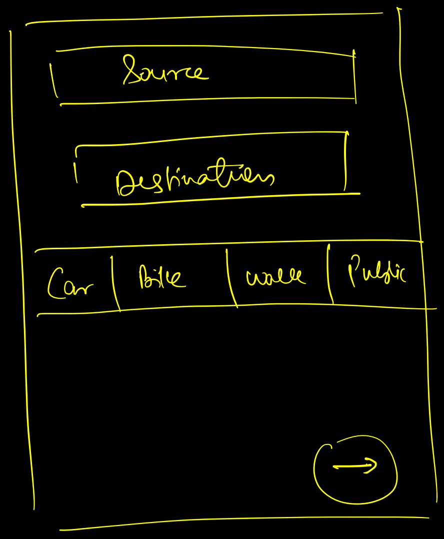
: Strategy Design Pattern ⎤
: Observer Design Pattern ⎦ ⇒ Behavioural Design patterns

Behavioural Design pattern :-

→ There are some special kind of behaviours that need to be implemented
↓
It helps with that

: Strategy Design Pattern :-



So when we choose | search for a path from location A to B in Google maps, there are multiple ways of transport, for each mode of transport there may be different ETA & path.

```
GoogleMaps {
        findPath( source, dest, mode) {
            if( mode == car) {
                                        logic for finding
                                        path via car
            } else if( mode == bike) {

            } else if( mode == walk) {

            } else if(
```

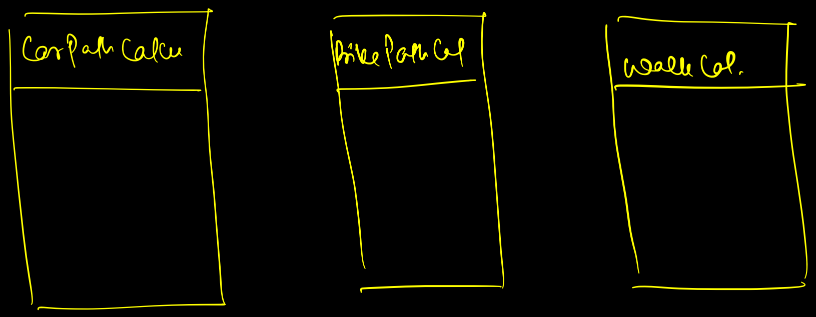
violated
OCP & SRP

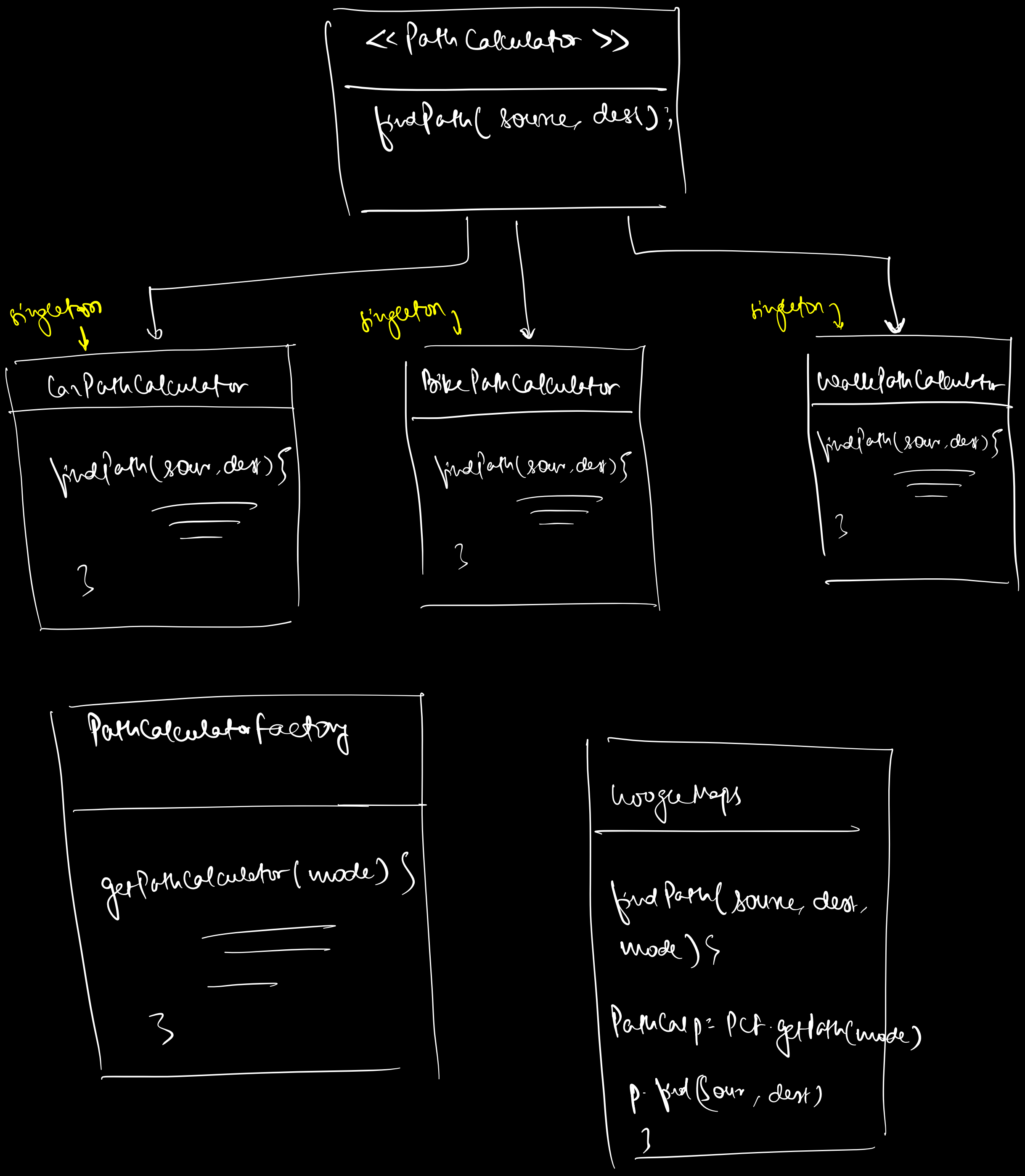
algos

When there are multiple ways to do something, often we might see violation of SRP/OCP, because

People might take care of the different ways by using multiple IF/ELSE blocks.

: Rather than implementing behaviour in a method, implement that in a separate class. For every unique way, create a particular class.



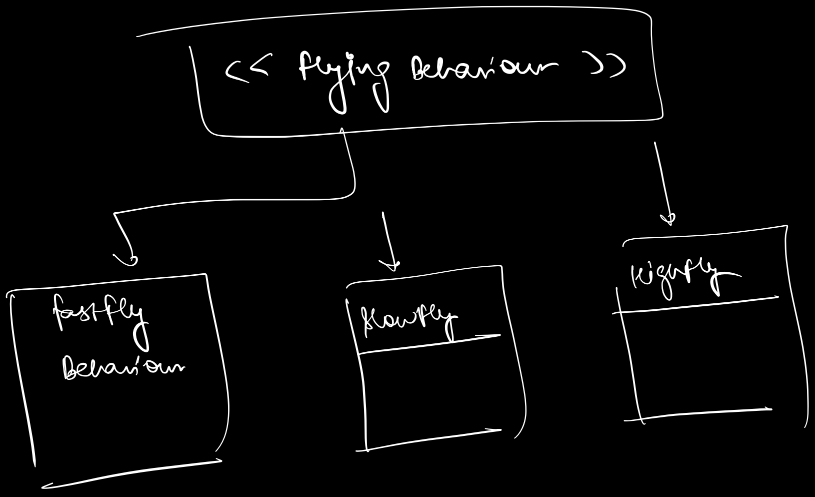


<< Path Calculator >>
findPath(source, dest);
singleton
singleton
singleton
CarPathCalculator
findPath(sour, dest){
}
BikePathCalculator
findPath(sour, dest){
}
WalkPathCalculator
findPath(sour, dest){
}
PathCalculatorFactory
getPathCalculator(mode) {
}
GoogleMaps
find Path(source, dest,
mode){
PathCal p = PCF.getPath(mode)
p.find(Sour , dest)
}

```
PathCalculator factory

CarPathCal c = CarPathCal.getInstance();

BikePathCal b = BikePathCal.getInstance();

walkPathCal w = walkPathCal.getInstance();

    getPathCalculator( mode) {

        if( mode = = car) {

            return c;

        else if( mode = = bike) {

            return b;

        else
            return w;
```

each type of calculator is a singleton class

Strategy → [multiple ways to do something, choose a particular way]



Asgn

Rain water trapping

		TC	SC
(I) —	BF	O(N2)	O(1)
(II) —	middle —	O(N)	O(N)
(III) —	best —	O(N)	O(1)

* Creat a strategy to solve the Rain water trapping problem in any way