Single Responsibility principle Tuesday Morch 3 ENSF 409, 2020 Open-Closed principle. Software Engineering Liskovs substitution principle Best practices Interface Segregation Dependency Inversion Single Responsibilier A class should only have one Responsey

Tunit sor a method

Tunit should only have one reason

to chance. over "5" help?

Thereasing cohesson reduces dependency.

Therease coupling

Tessing. How does "s" help? > Tegfing -) a class with one kerponsibility will have tener kerr case. > Bester organization

Open-Closal principle Classes should be open for extension, but closed to moditication. Lo By tollowing this principle (OC) we stop modifying existing classes, and don't cause potential Bugs! in an application that was working line betwee we tried to change it! now I want my app to a support motorbikes. class Bike { int id;
String make;
String make;
String engine Type, X = OC principle Instead of adding angine type, I can extend Bike. : According to oc principle, I proceed as follows

(2)

Bike Motor Bike engineType. class Employee ? Artista intlemelbrety feit | 2- part true 13- Com. 'salam. Nouthe salary; void calchalany (int type) { if (+ype == 1) { calcFull Time () else (+ype == 2){ Cal(Part Time L)

(3)

wasnest? Employer abstruct Calchay () E. ComBuployee culchay () calcpay () FulltimeEmp cal(Pay() Comfary Emp Ralchay () This design is closed Box to modification, and open to extention

Liskov's substitute and evaluate in an appliasion of class B a substee subclass of class B, then we should be able to replace B with A without disrupting the Nehmin of our program.

A A

Inheritance is Mohemanic it the parent is NOT general enough for its children.

In this case, the children may end up with methods/ variables that they so Not need!

(5)

(Cabquery) Animal makeSound() GoldFish 1209 makesound () makesound()> make Sand () Vuil makesond () void make Soud () {

Void make Soud () {

Void make Soud () {

Print ("Meon");

Print ("Bork");

}

Silly! The problem in the above example is there Arimal is NOT General enough. what is the solution?

-> Change the inheritance tree! Animal should be more general 40 gounds.

Solution (Carmet) Ansma cat () Silent Animal (calmond) Noisy Animal makeSond () ColdFish eart() ear () nog make Soul () makeSound() posh Car and Poor must implement: eas() and make Sound()

0

Intertale segregations «Interpace Pranable por ovan() Color Black And White Circle Rectungle drov () colorl) E Rus I Som () dungt color() need color! Solution ( Introduce ) «Interface» Promable Colorable Colorci drow () BlackAndWhiteCircle Rectaingle drov () draw() color ()

8

Longer Entertaces should be split into smaller ones. This army, we can engure these
the purplementing classes only need to be concerned
about the methods they need. public Torter face Bear (); void washTheBear (); void teed The Bear (); Too balky! preat into Void perthebear (); publi Enterbace Bear (eaner () { void wash The Bear (); papl: (Intertace Bear Feeder () { void teed The Bear (); Parlic Intubace. Beer Petter () & Void PetThe Ber ();

9)

Class Bear Cores implements Bear Cleaner, Bear Feeder & 11 implement Both methods Class Bearloner void wighthe Bear () { Con implement comply when it Soid teed The Bear () { reeds. CrayPason implements Bearletter { void perthe Bear () {

Depondency Inversion Retens to decoupling of softmare makeles. Instead ob high-level modules depending on low-level ones, Both Sepend on abstraction. This is much more Complet than the solution nelow Melanteray Injection Standard key pound WideScreen Moniter U Design Patter

(II)

In the solution above, class Computer is now depending on abstraction. you on change the type of Theonitor that the computer object Uses dynanically at runtime, without Changing the class Computer. By making? classes depend on abstruction you ensure that class is chosed for ... modification, but ofen to extension