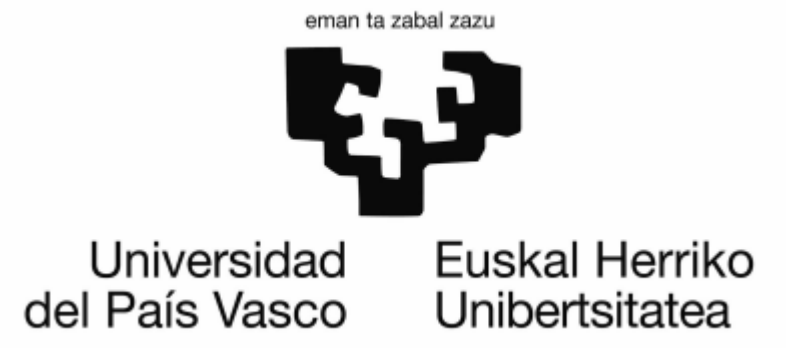

5. LABORATEGIA

SOLID printzipioak

31 DE OCTUBRE DE 2021

EHU-UPV

Software Ingenieritza 2

Aurkibidea

SARRERA

Dokumentu honetan, SOLID printzipioen laborategian egindako ariketak agertuko dira. Bost printzipio ezberdin agertuko dira. Atal bakoitzean galdera batzuen erantzuna agertuko dira urdinez. Kodea ez da dokumentu honetan agertuko, github helbidetik hartu beharko da kodea aldaketak ikusteko.

Github errepositorioaren helbidea:

JonOchoadeAlda/SolidLaborategia: Solid laborategia (github.com)

Ireki-Itxia Printzipioa (OCP)

Galderak:

1. Zer gertatuko litzateke Sheet klaseari “Diamond” gehituko bagenio? Sheet klaseak OCP printzipioa betetzen du?. Erantzuna justifikatu. Betetzen ez badu, behar diren aldaketak egin printzipioa betetzeko.

   Sheet klaseari “diamond” gehituz gero vector berri bat egin beharko genuke pere addDiamond metodoarekin eta drawFigure metodoa ere aldatu beharko genuke. OPC printzipioa ez du betzen aipatutakoagatik, aldaketak egon daitezkeen puntuan diseinu itxia dagoelako. OPC printzipioa betetzeko “figura” interfaze bat sortu beharko genuke etorkizunean beste figura batzuk gehitu nahi baditugulako.

   2. Aplikazio bat sortu behar diren klase guztiekin, eta programa nagusi batean, drawFigures() metodoari deitzen dion programa bat sortu, bi circle, square bat eta diamond objektu batekin.

   3. Egin duzun aldaketa errefaktorizazio bat dela kontsideratzen duzu? Justifikatu erantzuna.

   Errefaktorizazioa dela esango nuke barne egitura aldatu dugulako kanpoko portaera aldatu gabe.

   4. Figure bakoitzak azalera bat edukiko balu (hau da, getArea()), eta Sheet batean dauden irudi guztien azalera kalkulatu nahi bagenu, nola osatuko zenuke klaseak. Programa nagusia osatu, eskakizun hau frogatzeko.

Erresponsabilitate bakarreko printzipioa (SRP).

Eskatzen da:

1.Aplikazioa errefaktoriazatu, erresponsabilitate bakoitza klase isolatu batean geratu dadin. Adierazi zein aldaketa egin beharko litzateke fakturaren dedukzioa (hau da, billDeduction) fakturaren arabera kalkulatuko balitz:

Dedukzioaren kalkulua egiteko klase bat sortuko genuke eta Vata kalkulatzeko beste bat.

2. Zein aldaketa egin beharko litzateke VAT-a %16-tik %18-ra aldatuko balitz?

Vat klasearen portzentaia aldatuko genuke. Beste aukera bat OCP printzipioa erabiltzea da, Vat portzentai desberdinak erabiliko badira aldi berean

3. Zein aldaketa egin beharko litzateke, 0 kodea duten fakturei VAT-a aplikatzen EZ bazaie.

TotalCalc metodoan if bat jarri eta kodea 0 denean kalkulatu dugun vata ez gehitu.

Liskov printzipioa (LSK)

Galderak:

1. Programa batean, Project bat sortu 3 fitxategi mota desberdin batzuekin, eta jarraian bere metodoak exekutatu.

2. Project klaseak OCP printzipioa betetzen du?. Justifikatu erantzuna.

Ez du betetzen Project klasea klase abstraktu edo interfaze batekin erlazioa eduki beharko lukeelako eta es ProjectFile klasearekin zuzenean. Horretarako File klase abstraktu bat sortuko dugu.

3.Project klaseak LSK (Liskov) printzipioa betetzen du?. Justifikatu erantzuna.

Ez du betezen ReadOnlyProjectFile klaseak storeFile metodoa inplementatu behar duelako baina file hauek ezin dute portaera hori izan. Konpontzeko herentzia ezabatu behar da eta komunean duten metodoak interfaze batera mugitu Fload interfazera. Eta FStore interfazea sortuko dugu storeFile metodoa heredatu dezaketen klaseetarako.

4. Errefaktorizatu aplikazioa OCP edo LSK betetzen ez badu.

Dependentzi inbertsioaren printzipioa (DIP)

Srp printzipioan erabili dudan klaseak erabili ditut zeuden klaseak konpilazio errore asko ematen zutelako eta Bill klaseak DIP printzipioa ez duelako betetzen.

Galderak:

1. DIP printzipioa betetzen du. Justifikatu erantzuna.

   Ez du betetzen, Bill klasea zuzenean Deduction eta Vat klasearekin dependentzia dauka, dependentzia hori interfazeen bidez konponodu behar da IVAT eta IDeduction interfazeak sortuz. Bill klasean totalCalc() metodoari parametro moduan bidali beharko ziren deduction eta vat objetuak.

2. Errefaktorizatu kodea betetzen ez badu.

Interfaze Bananduaren Printzipioa (ISP)

Galderak:

1. Zer informazio behar dute EmailSender eta SMSSender klaseek egin behar duten funtzionalitatea betetzeko, eta zer informazio jasotzen dute? ISP printzipioa bortxatzen dutela uste duzu?.

   ISP printzipioa bortxatzen du, informazio gehiegi jasotzen dutelako, pertsona objetua pasatzen denez parametro moduan bere metodo guztiak erabil ditzake baina ez ditu behar.

2. Aurreko klaseak errefaktorizatu (EmailSender eta SMSSender), Person parametroa aldatuz interface batengatik (klase bakoitzak interfaze desberdina). Interface bakoitzak klase bakoitzak behar duen metodoez osatuta egongo da bere eginkizuna betetzeko, hau da, mail bat edo sms bat bidaltzeko. Person klasea ere aldatu.

3. GmailAccount klasea osatu. Klase honetako objektuek (aldaketa batzuekin) EmailSender klasera bidaltzeko ahalmena izan beharko lukete baina ez SMSSender klasera.