Diseinu patroia

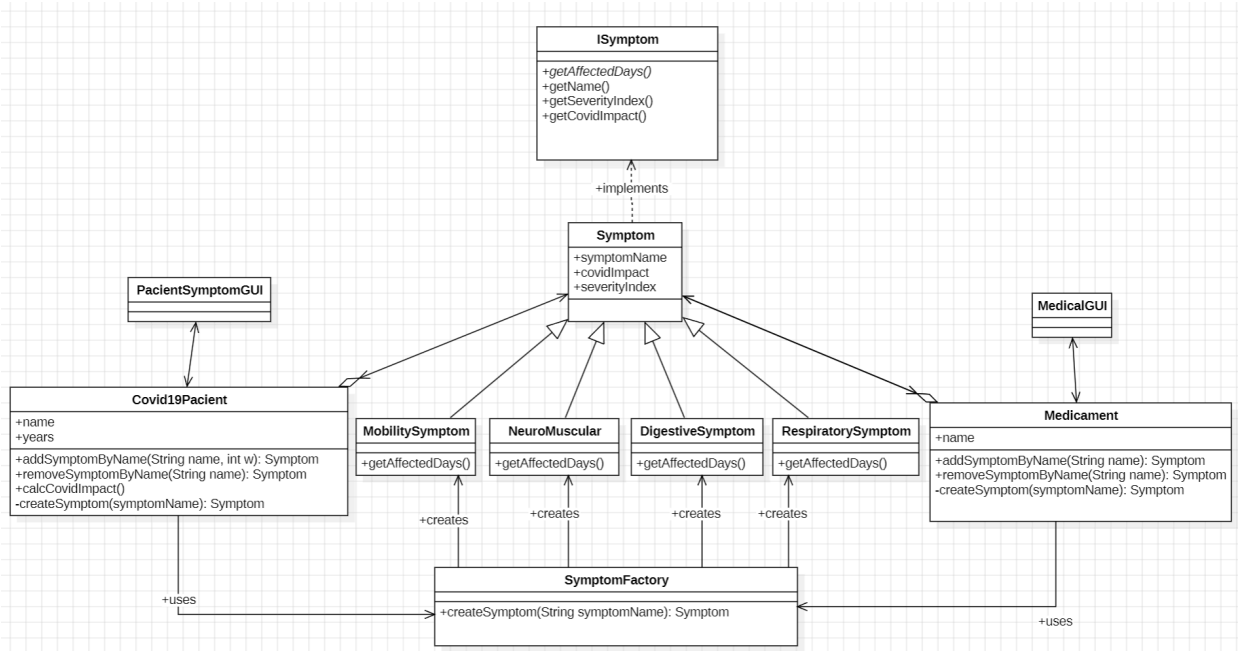
Github esteka:https://github.com/qianLi31/labpatterns.git
Talde Kideak: Iker Sada eta Qian Lizhang.

Simple Factory

1. Zer gertatzen da, sintoma mota berri bat agertzen bada (adb: MovilitySymptom)?
OCP printzipioa bortxatzen du, zereb Medicament eta Covid19Pacient klasea aldatu behar duelako.
2. Nola sortu daiteke sintoma berri bat orain arte dauden klaseak aldatu gabe (OCP printzipioa)? Abstrakzioa sortu.
3. Zenbat erresponsabilitate dauzkate Covid19Pacient eta Medicament klaseak (SRP printzipioa)? Pazientearen datuen kudeaketa (name, age) eta Sintomen zerrenda kudeaketa (add, remove, get).

Eskatzen da:
1. Aplikazioaren diseinu berri bat egin (UML diagrama) Simple Factory patroia aplikatuz, aurreko ahultasunak ezabatzeko. Jarraian deskribatu testu batean egin dituzun aldaketak.

Sintoma berri bat gehitzeak Factory patroiaren abantaila nagusia erakusten du. Jatorrizko diseinuan, sintoma berri bat gehitzeko Covid19Pacient eta Medicament klaseetako createSymptom metodoak aldatu behar genituen. Factory patroiarekin, aldaketa bakarra egin behar da.



2. Aplikazioa inplementatu, eta "mareos" sintoma berria gehitu 1 inpaktuarekin.

```
List<String> impact1 = Arrays.asList("nauseas", "vómitos", "congestión nasal","diarrea",
        "hemoptisis","congestion conjuntival", "mareos");  // GEHITU "mareos"
List<Double> index1 = Arrays.asList(5.0, 4.8, 3.7, 0.9, 0.8, 2.0); // GEHITU "mareos"
List<String> digestiveSymptom=Arrays.asList("nauseas", "vómitos","diarrea");
List<String> neuroMuscularSymptom=Arrays.asList("fiebre", "astenia", "cefalea",
        "mialgia","escalofrios");
List<String> respiratorySymptom=Arrays.asList("tos seca","expectoracion","disnea","dolor de garganta",
        "congestión nasal","hemoptisis","congestion conjuntival");
List<String> mobilitySymptom=Arrays.asList("mareos");  // "mareos" sintoma berria
```

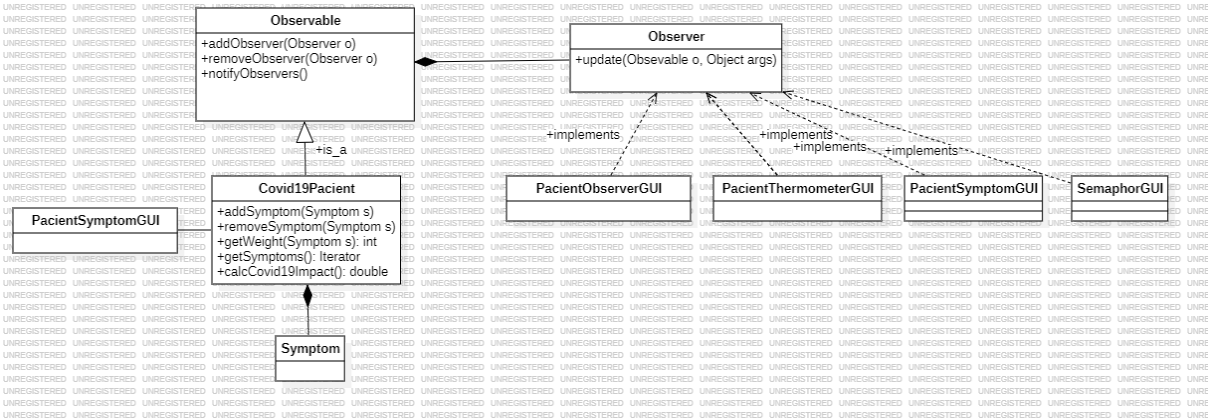
3. Nola egokitu daiteke Factory klasea, Covid19Pacient eta Medicament erabiltzen dituzten Symptom objektuak bakarrak izateko? Hau da, sintoma bakoitzeko objektu bakar bat egon dadin. (x sintoma sisteman badaude, x objektu Symptom soilik)

Sintoma objektuen instantzia bakarra lortzeko Singleton patroiaren bariante bat aplikatu dugu. Ez da klase Singleton tradizionala, baizik eta "Sintoma Pool" bat, non sintoma bakoitzaren instantzia bakarra gordetzen den.

Observer Patroia

Eskatzen da:

1. Aplikazioaren UML diagrama hedatuta egin dituzun aldaketan aurkeztuz.



Observer interfazeari gehitu diogu bi GUI berriak, PacientSymtpmGUI eta SempahorGUI.

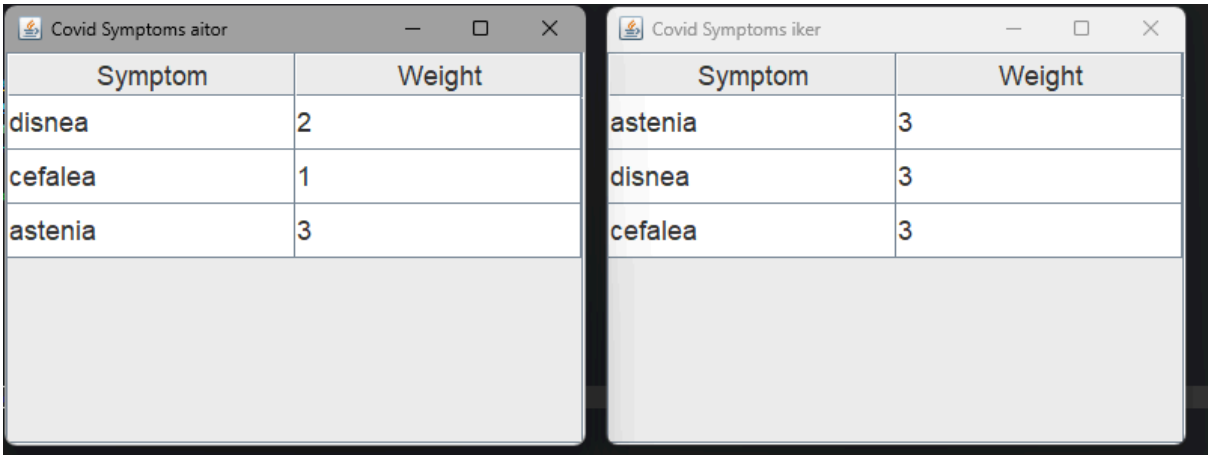
2. Aplikazioaren inplementazioa.

Jarriko det Main implementazio, dena jartzea oso handia delako, bestela GitHubean dago dena:

```
public static void main(String[] args) {
        Observable pacient=new Covid19Pacient("aitor", 35);
        new PacientObserverGUI (pacient);
        new PacientSymptomGUI ((Covid19Pacient) pacient);
        new PacientThermometerGUI(pacient);
        new SemaphorGUI(pacient);
```

Adapter Patroia

Eskatzen da:
• Behar den kodea gehitu Covid19PacientTableModelAdapter klasean eta aplikazio exekutatu emaitza konprobatuz..
• Beste paziente bat gehitu sintoma batzuekin, eta aplikazioa exekutatu bi taulak agertzen direla konprobatuz.



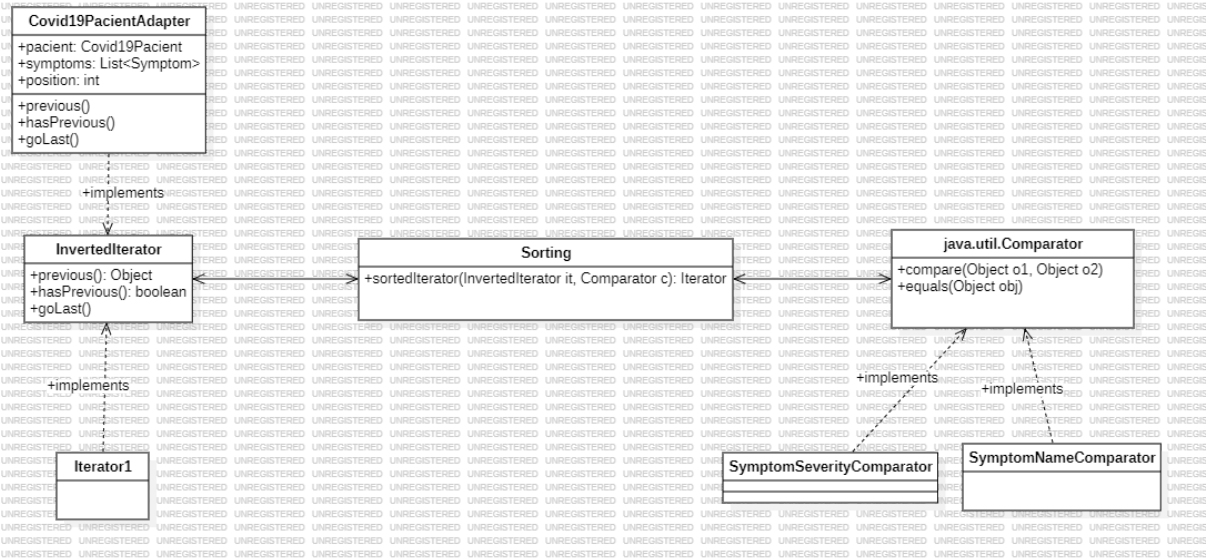
• (hautazkoa). Nola gehituko zenuke taula lehio bat (ShowPacientTableGUI)aurreko observer ariketan, paziente bateri sintoma berri bat gehitzen zaion bakoitzean bere taula leihoa eguneratzeko?

```
public class Covid19PacientTableModelAdapter extends AbstractTableModel implements Observer {

   @Override
    public void update(Observable o, Object arg) {
        fireTableDataChanged();
    }

//Beste metodoak
}
```

Adapter Iterator eta Patroiak

Eskatzen da:
Programa Nagusi batean 5 Symptom-a duen Covid19Pacient bat sortu, eta jarraian Sorting.sortedIterator metodoa erabiliz, bere sintomak inprimatu lehendabizi symptomName ordenatuaz, eta jarraian severityIndex ordenatuaz. Covid19Pacient eta Sorting klasetan EZIN DA EZER ALDATU. Horretarako hurrengo pausoak jarraitu:

1- UML diagrama aplikazioaren diseinuarekin.



2- Comparator interfazea inplementatzen dituzten bi klase definitu elementuak symptomName eta severityIndex ordenatzen dituztenak hurrenez hurren.

Bi konparatzaile desberdin inplementatu ditugu:

SymptomNameComparator: Sintomen izenen arabera ordenatzen du.

SymptomSeverityComparator: Sintomen larritasun indizearen arabera ordenatzen du.

```
public class SymptomNameComparator implements Comparator<Object> {
  @Override
  public int compare(Object o1, Object o2) {
     Symptom s1 = (Symptom) o1;
     Symptom s2 = (Symptom) o2;
     return s1.getName().compareTo(s2.getName());
  }
}
```

```
public class SymptomSeverityComparator implements Comparator<Object> {
  @Override
  public int compare(Object o1, Object o2) {
     Symptom s1 = (Symptom) o1;
     Symptom s2 = (Symptom) o2;
     return Double.compare(s1.getSeverityIndex(), s2.getSeverityIndex());
  }
}
```

3- Covid19Pacient klasea InvertedIterator interfazera egokitzen duen klase adaptadorea sortu. Gogoratu metodo eraikitzaile egokia sortzea pazientearen informazioa bidaltzeko.

Covid19PacientInvertedAdapter klaseak bi funtsezko egokitzapen egiten ditu:

Datu egituraren egokitzapena: Covid19Pacient-en Set<Symptom> datu egitura List<Symptom>-era bihurtzen du.

Interfazearen egokitzapena: InvertedIterator interfazearen metodoak inplementatzen ditu (previous(), hasPrevious(), goLast()).

```
public class Covid19PacientInvertedAdapter implements InvertedIterator {
  private Covid19Pacient pacient;
  private List<Symptom> symptoms;
  private int position;
  public Covid19PacientInvertedAdapter(Covid19Pacient p) {
     this.pacient = p;
     // Si getSymptoms() devuelve Set<Symptom> o Collection<Symptom>
     this.symptoms = new ArrayList<>(p.getSymptoms());
     goLast();
  }
```

+... @Override beste metodoak

4- Programa nagusi batean, Covid19Pacient objektu bat sortu 5 sintomekin, eta Sorting.sort metodoari bi aldiz deitu, sortu duzun CovidPacient klase adaptadorea eta Comparator inplementatu dituzun bi konparadorearekin. Bukatzeko emaitza inprimatu pantaiatik.

Programa nagusiak honako pausoak jarraitzen ditu:

Pazientea eta sintomak sortu: 5 sintoma desberdinekin.Bi ordenazio prozesu exekutatu:

-Lehenengoa izenaren arabera ordenatzen du.

-Bigarrena larritasunaren arabera ordenatzen du.

```
+---- Izena order ----+
astenia
cefalea
disnea
fiebre
tos seca

+---- Severity order ----+
disnea
cefalea
astenia
tos seca
fiebre
```