TP de révision

Romain MAURICE

Record Apartment

```
import java.util.List;
import java.util.Objects;
import java.util.stream.Collectors;

public record Apartment(int area, List<String> personnes) implements Housing{

    public Apartment {
        Objects.requireNonNull(List.copyOf(personnes));
        if(area<0)
            throw new IllegalArgumentException();
    }

    @Override
    public String toString() {

        return personnes.stream().collect(Collectors.joining(", ","Apartment "+area+"
m2 with "," "+efficiency()));

    }

    @Override
    public double price() {
        return 20*personnes.stream().count();
    }

    @Override
    public double efficiency() {
        if(personnes.stream().count()>=2)
            return 1.0;
        else
            return 0.5;
    }

}
```

Class AssetManager

```
import java.util.ArrayList;
import java.util.List;
import java.util.Objects;
import java.util.stream.Collector;
import java.util.stream.Collectors;
```

```java
public class AssetManager {

    private ArrayList<Housing> list;

    public AssetManager() {
        this.list = new ArrayList<>();
    }

    public void add(Housing housing) {
        Objects.requireNonNull(housing);
        list.add(housing);
    }

    public double profitPerNight() {
        return list.stream()
                .mapToDouble(i->i.price())
                .sum();
    }

    public List<Housing> lowestEfficiency(double value){
        return list.stream().filter(i->i.efficiency()<value).toList();
    }

    public void remove(double value) {
        if(value<0 || value>1)
            throw new IllegalArgumentException();
        list.removeIf(i->i.efficiency()<value);
    }

    /*@Override
    public String toString() {
        StringBuilder sb = new StringBuilder();
        for(var i : list) {
            sb.append(i.toString());
            sb.append("\n");
        }
        return sb.toString();
    }*/

    @Override
    public String toString() {
        return list.stream().map(i->i.toString()).collect(Collectors.joining("\n"));
    }

}
```

Class Hotel

```java
public final class Hotel implements Housing{
    private final int rooms;
```

```
    private final double efficiency;

    public Hotel(int rooms, double efficiency) {
        if(rooms < 0)
            new IllegalArgumentException();
        if(efficiency < 0 || efficiency > 1)
            throw new IllegalArgumentException();

        this.rooms = rooms;
        this.efficiency = efficiency;
    }

    @Override
    public String toString() {
        return    "hotel "+rooms+" rooms "+efficiency;
    }

    @Override
    public double price() {
        return 100*rooms*efficiency;
    }

    @Override
    public double efficiency() {
        return efficiency;
    }
}
```

Interface Housing

```
public sealed interface Housing permits Apartment, Hotel {

    public abstract double price();
    public abstract double efficiency();
}
```

Main

```
import java.util.ArrayList;
import java.util.List;

public class Main {
    public static void main(String[] args) {
          /*var hotel = new Hotel(5, 0.75);
          System.out.println(hotel);  // Hotel 5 rooms 0.75

          var apartment = new Apartment(30, List.of("Bony", "Clyde"));
          System.out.println(apartment); // Apartment 30 m2 with Bony, Clyde 1.0

          var list = new ArrayList<String>();
          list.add("Bob");
```

```
        var apartment2 = new Apartment(50, list);
        list.remove("Bob");//ne doit pas marcher -> liste immuable
        System.out.println(apartment2);  // Apartment 50 m2 with Bob 0.5
        */
        var hotel3 = new Hotel(5 , 0.75);
        var apartment3 = new Apartment(30, List.of("Bony", "Clyde"));
        var manager = new AssetManager();
        manager.add(hotel3);
        manager.add(apartment3);
        System.out.println(manager.profitPerNight());  // 415
        System.out.println(manager); // affiche
        // Hotel 5 rooms 0.75
        // Apartment 30 m2 with Bony, Clyde 1.0

        System.out.println(manager.lowestEfficiency(0.8));  // [Hotel 5 rooms 0.75]
}


        manager.remove(0.8);
        System.out.println(manager);  // affiche
        // Apartment 30 m2 with Bony, Clyde 1.0
    }
}
```