CS312 – Programmation Objet

TP3 : Gestion de carte de restaurant

L'objectif de ce TP est de vous familiariser avec l'utilisation des interfaces et des classes abstraites dans le cadre du développement d'une application destinée à automatiser la gestion des menus et des commandes des restaurants. L'objectif est, dans un premier temps, de pouvoir créer la carte d'un restaurant à l'aide de plats et de boissons. La possibilité de créer de menus, à partir des plats de la carte, est également offerte. L'application permet ensuite de définir des commandes.
Une première version de la définition des classes de l'application (éventuellement incomplètes) est donnée en annexe.

Les plats et boissons

Nous représentons les plats et les boissons du restaurant à l'aide des classes `Consommable`, `Boisson`, `Plat`, `Entrée`, `PlatPrincipal`, `Dessert`.

Question 1
Modifier le code de sorte que `Consommable` soit une interface (et non une classe).

```
public interface Consommable {
    public String getNom();
    public int getPrix();
}
```

Modifier en conséquence les classes `Plat` et `Boisson`.

Question 2
Réaliser une classe de test `TestRestaurant` avec une méthode `main` que vous compléterez au fur et à mesure que vous répondrez aux questions pour tester vos solutions.

Les menus

Un menu correspond à une offre groupée de plats avec une boisson, proposée à un prix inférieur ou égal à la somme des prix des produits commandés séparément. Il est matérialisé par la classe `Menu`.

Question 3
(i) Compléter le constructeur de la classe `Menu`.
(ii) Compléter la méthode `toString` de cette même classe afin qu'elle affiche la liste des plats et la boisson du menu (par exemple : *"Menu composé de Salade verte, Pizza Reine, Tiramisu, Eau, au prix de 15 euros"*).

Nous souhaitons vérifier, à l'intérieur de la classe `Menu`, que le prix fixé pour un menu est bien inférieur ou égal au prix des plats et boisson pris séparément. Pour cela, on apportera les modifications suivantes:

Question 5
Ajouter à la classe `Menu` la méthode privée

```
    private boolean verifPrixMenu().
```

Cette méthode retourne une valeur vraie uniquement si *la somme des prix des items composant un menu est supérieure ou égale au prix du menu qui doit lui-même être strictement positif.*

La carte

La carte du restaurant contient l'ensemble de plats et boissons proposés par le restaurant classés par type (entrées, plats principaux, desserts, boissons) ainsi que les différents menus. La classe `Carte` joue ce rôle. On peut constater, en lisant son implémentation incomplète, qu'au moment d'ajouter un item dans la carte, on vérifie qu'un item du même nom n'y apparaît pas déjà (afin d'éviter qu'un plat nommé "Tiramisu" soit inséré tandis qu'une entrée du même nom figure déjà dans la carte).

Question 6

Réaliser la méthode `verifCarte` qui effectue cette vérification.

Une autre vérification est effectuée au moment de l'insertion des menus dans la carte : tous les items (plats, boissons) composant le menu doivent figurer déjà dans la carte.

Question 7
Réaliser la méthode `verifMenu` qui effectue cette vérification.

Les commandes

Une commande est constituée d'un ensemble d'items, plats et boissons. La classe Commande permet de représenter ce concept. Pour calculer le prix d'une commande, notre application vérifie automatiquement si un sous ensemble de ces items correspond à un menu de la carte, auquel cas, ils sont facturés au prix du menu.
Par exemple, considérons une carte qui comporte:

- une entrée Salade verte à 4 €
- une entrée Salade composée à 6 €
- un plat principal Pizza Reine à 9 €
- un plat principal Pizza Margarita à 8 €
- un plat principal Spaghetti à la Bolognaise à 25 €
- un dessert Tiramisu à 4 €
- une boisson Eau gratuite
- un menu *Salade verte, Pizza Reine, Tiramisu, Eau* au prix de 15 €.

Un client commandant une Pizza Reine, une Pizza Margarita, une Salade verte et un Tiramisu avec de l'Eau devrait payer:

- 8 €pour la Pizza Margarita
- 15 € pour le menu composé de la Salade verte, la Pizza Reine, le Tiramisu et l'Eau

soit 23 € (et non pas 25 € qui est la somme des prix des items).

Au cas où la même combinaison de plats peut apparaître dans plusieurs menus, on privilégie systématiquement celui qui apparait en premier dans la liste des menus de la carte.

Question 8
Ajouter la méthode `public int calculerPrixCommande(Commande c)` à la classe `Carte` calculant le prix d'une commande selon le principe ci-dessus .

La diététique

Nous souhaitons apporter à l'application une dimension diététique en permettant aux consommateurs de commander des menus dont ils connaissent les informations nutritionnelles. Pour cela, nous définissons l'interface suivante:

```
public interface Nutrition {    // S'applique à une portion moyenne
    public int getKcal();       // nombre de Kcal
    public float getGlucides(); // grammes de glucides
}
```

Question 9
Modifier en conséquence les classes `Plat`, `Boisson` et leurs sous-classes.

Question 10
Ajouter la méthode `public void proposerMenu(int Kc, int epsilon)` à la classe `Carte`, affichant à l'écran tous les menus qui totalisent un nombre de Kcal égal à Kc (à epsilon Kcal près).

Lecture à partir d'un fichier

Question 11
En utilisant les classes `BufferedReader` et `FileReader`, ajouter à la classe `Carte` un nouveau constructeur ayant un paramètre qui est le nom d'un fichier texte contenant la totalité des informations sur les plats et menus (dans une syntaxe que vous définirez).

ANNEXE

```
abstract public class Consommable {
    private String nom;
    private int prix;  // en cents d'euros

    public Consommable(String nom, int prix){
        this.nom = nom; this.prix = prix;
    }
    public String getNom(){
        return this.nom;
    }
    public int getPrix(){
        return this.prix;
    }
}

public class Boisson extends Consommable{
    private int volume; // en centilitres

    public Boisson(String nom, int prix, int volume){
        super(nom, prix);
        this.volume = volume;
    }

    public Boisson(String nom, int volume){
        this(nom, 0, volume);
    }

    public int getVolume(){
        return volume;
    }
}

public class Plat extends Consommable{
    public Plat(String nom, int prix){
        super(nom, prix);
    }
}

public class Entrée extends Plat {
    public Entrée(String nom, int prix) {
        super(nom, prix);
    }
}

public class PlatPrincipal extends Plat {
    public PlatPrincipal(String nom, int prix) {
        super(nom, prix);
    }
}

public class Dessert extends Plat {
    public Dessert(String nom, int prix) {
        super(nom, prix);
    }
}
```

```java
public class Menu {
    private ArrayList<Consommable> items;
    private int prix; // en cents

    public Menu(int prix, Entrée e, PlatPrincipal p, Dessert d, Boisson b) {
        // Compléter
    }

    public ArrayList<Consommable> getItems(){
        return this.items;
    }

    public int getPrix(){
        return this.prix;
    }

    public String toString(){
        String message = "Menu composé de ";
        // Compléter
        message += "au prix de " + this.prix/100 + " euros";
        return message;
    }
}
```

```java
public class Carte {
    private ArrayList<Consommable> entrées;
    private ArrayList<Consommable> platsPrincipaux;
    private ArrayList<Consommable> desserts;
    private ArrayList<Consommable> boissons;

    private ArrayList<Menu> menus;

    public Carte() {
        entrées = new ArrayList<Consommable>();
        platsPrincipaux = new ArrayList<Consommable>();
        desserts = new ArrayList<Consommable>();
        boissons = new ArrayList<Consommable>();
        menus = new ArrayList<Menu>();
    }

    public void addEntrée(Entrée e){
        if (verifCarte(e)) this.entrées.add(e);
    }

    public void addPlatPrincipal(PlatPrincipal p){
        if (verifCarte(p)) this.platsPrincipaux.add(p);
    }

    public void addDessert(Dessert d){
        if (verifCarte(d)) this.desserts.add(d);
    }

    public void addBoisson(Boisson b){
        if (verifCarte(b)) this.boissons.add(b);
    }

    public void addMenu(Menu m){
        if (verifMenu(m)){
            this.menus.add(m);
```

```
        }
    }

    public ArrayList<Consommable> getEntrées(){
        return this.entrées;
    }

    public ArrayList<Consommable> getPlatsPrincipaux(){
        return this.platsPrincipaux;
    }

    public ArrayList<Consommable> getDesserts(){
        return this.desserts;
    }

    public ArrayList<Consommable> getBoissons(){
        return this.boissons;
    }

    // Vérifie que les plats et boissons du menu sont bien dans la carte
    private boolean verifMenu(Menu m){
        // Compléter
    }

    // Vérifie qu'il n'y a pas d'homonymes dans la carte
    private boolean verifCarte(Consommable c){
        // Compléter
    }

    /* Calcule le prix de la commande. A priori, ce prix est la somme des prix
      des items SAUF si une partie de ces items constituent un menu; dans ce cas,
      le tarif menu s'applique. */
    public int calculerPrixCommande(Commande c){
        // Compléter

    }
}
```

```
public class Commande {
    private ArrayList<Consommable> itemsCommandés;

    public Commande() {
        this.itemsCommandés = new ArrayList<Consommable>();
    }

    public void addItem(Consommable c){
        this.itemsCommandés.add(c);
    }

    public ArrayList<Consommable> getItemsCommandés(){
        return this.itemsCommandés;
    }
}
```