TD 3 Les constructeurs, les variables statiques et l'interaction des variables

Le but de ce TD sera de simuler une pizzeria. Une pizza sera assemblée à partir d'une pâte, une sauce, et au plus trois ingrédients (toppings). La pâte pourrait être fine, croustillante ou épaisse. Nous allons avoir une sauce tomate, une sauce à la crème ou une sauce barbecue. Les ingrédients auront des noms (per exemple : œuf, champignons, jambon, etc.) et une taille de portion. Pour garder en tête toutes les méthodes de toutes les classes dans lesquelles on va travailler, je vous conseille de regarder les diagrammes de chaque classe qu'on va utiliser.

Exercice I : Classes et constructeurs

Prenons une classe Pate, qui inclue le code suivant :

```
public class Pate {
	private String type;
	private char taille;

	public String getType() {
		return this.type;
	}

	public char getTaille() {
		return this.taille;
	}

	public String toString() {
		return("Pate[" + this.taille + "; " + this.type + "]");
	}
}
```

- Dans une classe TD3, dans une méthode principale (main) on a ces deux lignes de code :

  ```
  final Pate pateCrousti = new Pate();
  System.out.println(pateCrousti);
  ```

  Est-ce que ce code compile ? Qu'est-ce qui se passe si on veut l'exécuter ?

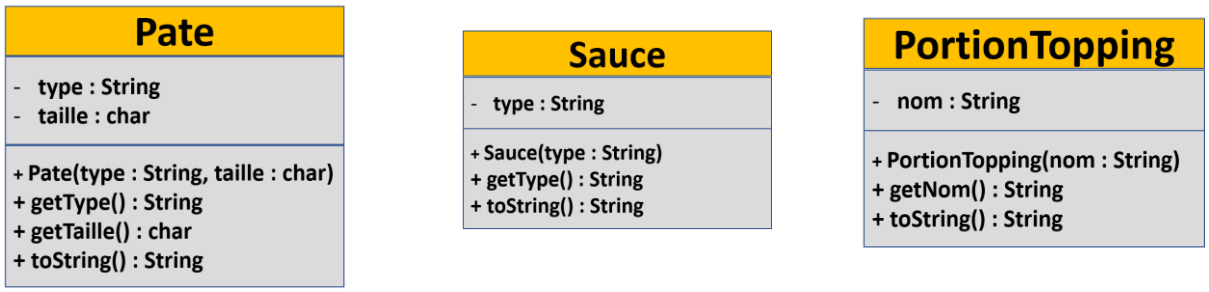
- Nous allons ajouter dans la classe Pate le constructeur suivant :

  ```
  public Pate(String type, char taille) {
  	this.type = type;
  	this.taille = taille;
  }
  ```

Est-ce que le code de l'exercice précédent compile ? Avec quel résultat ?

Exercice II : Construire des Pizzas

Nous allons partir sur les diagrammes de classe suivants :



Dans la classe `Pate` nous avons deux attributs : un attribut type de type `String`, qui peut prendre les valeurs `"croustillante"`, `"fine"`, `"epaisse"` ; ainsi qu'un attribut taille de type `char`, qui peut prendre les valeurs 'S', 'M', 'L'. Le constructeur met les valeurs des attributs aux valeurs mises en entrée. Les getters renvoient les attributs dont ils portent les noms. La méthode String toString() retourne le texte suivant :

`Pate[<taille> ; <type>]`

Dans la classe `Sauce` nous avons un seul attribut, notamment le type, qui peut prendre les valeurs : `"BBQ", "creme fraiche", "tomates".` Le constructeur de cette classe prend en entrée une valeur String qui sera assignée à l'attribut de la classe. Le getter retourne la valeur actuelle du type de la sauce. Finalement, la méthode `toString()` rend le texte suivant :

`Sauce[<type>]`

Dans la classe `PortionTopping` nous avons un attribut nom de type `String`. Le constructeur assigne à l'attribut la valeur en entrée. Le getter retourne la valeur actuelle de l'attribut nom. Finalement, la méthode `toString()` rend le texte suivant :

`Ingredient[<type>]`

Nous voulons construire une classe `Pizza` à partir de ces trois classes. On commence avec une classe `Pizza` qui aura les attributs suivants : un attribut pate de type `Pate`, un attribut sauce de type `Sauce`,

un attribut taille de type `char`, un attribut `toppings` de type `PortionTopping[]` et un attribut `prix` de type `double`.

- On calcule le prix d'une pizza à partir d'un prix de base (selon la taille) et selon le nombre de portions de topping. Le prix de base est : 8 euros pour une pizza de taille S, 10 euros pour une pizza M et 12 euros pour une pizza L. Les premières trois toppings sont gratuites ; puis, pour chaque topping supplémentaire on ajoute 50 centimes au prix.

  Ecrivez une méthode privée avec la signature double `calculerPrix()` qui rend le prix de la pizza actuelle.

- En utilisant votre méthode `calculerPrix()` écrivez un constructeur avec la signature `Pizza(Pate pate, Sauce sauce, PortionTopping[] toppings)` qui assigne les valeurs en entrée aux attributs : `pate`, `sauce`, `toppings` respectivement, qui met la taille de la pizza à la taille de la pâte et qui appelle la méthode `calculerPrix` pour assigner une valeur à l'attribut `prix`.

- Ecrivez une méthode `String toString()` qui retourne le texte suivant pour chaque pizza :

  ```
  Taille : <taille> ;
  Pate[<taille> ; <type>] ;
  Sauce[type] ;
  Toppings[<noms des toppings separes pas un espace>] ;
  Prix : <prix>
  ```

- Dans la classe TD3, dans une méthode principale, écrivez du code qui crée les prochaines pizzas :
  - Une pizza exotique de taille M, avec une pâte croustillante, une sauce à la crème fraîche et 3 toppings : jambon, ananas et du bleu.
  - Une pizza kebab de taille L, avec une pâte fine, une sauce BBQ et quatre toppings : kebab, emmental, paprika et olives
  - Une pizza Iberia de taille S, avec une pâte épaisse, une sauce tomates et quatre toppings : jambon sec, mozzarella, roquette et pesto vert.

- Supposons maintenant que dans la classe `Pizza` a des getters pour chaque attribut. Ecrivez du code qui vous permet d'afficher le nombre de toppings de la pizza exotique, le nombre et tailles de pates utilisées et du code qui vérifie si la taille de la pizza Iberia coïncide avec la taille de la pate qu'on a utilisée pour cette pizza.

Exercice III : Les attributs statiques

Dans cet exercice nous allons travailler sur les attributs statiques. Rappel : on en a parlé des attributs statiques lors des CM.

- Qu'est-ce qu'un attribut statique ?
- Une pizzeria veut estimer combien de portions de farine sont utilisées dans les pizzas qu'elle vend chaque jour. Pour faire cela il faut prendre en compte le nombre de pizzas vendues ainsi que leurs tailles. Nous allons ajouter un attribut statique `quantiteUtilisee` de type `int` aux attributs de la classe `Pate`. Cet attribut sera initialement mis à zéro.

  Nous allons modifier le constructeur tel que à chaque fois qu'on crée une `Pate`, on augmente la valeur de `quantiteUtilisee` de la façon suivante :
  - Si la pate est de taille `S`, on incrémente la valeur par 1
  - Si la pate est de taille `M`, on incrémente la valeur par 2
  - Si la pate est de taille `L`, on incrémente la valeur par 3

  De cette façon la pizzeria aura un nombre total de portions de farine utilisées.

  Ecrivez le nouveau constructeur de cette classe.

- Rappel : quelles sont les règles quant aux attributs statiques et les méthodes statiques ?

- Ecrivez une méthode `static int getQuantiteUtilisee()` qui retourne la valeur de l'attribut statique `quantiteUtilisee`.

- Dans la méthode main de la classe TD3 (qui aura les lignes de code dans l'exercice II.4) écrivez du code qui affiche la valeur de l'attribut `quantiteUtilisee`
  - En appelant une méthode de la classe `Pate` pour un objet du type `Pizza`
  - En appelant une méthode de la classe `Pate`, mais en n'utilisant aucun objet

- Dans la méthode main on a ces deux lignes de code. Est-ce qu'il compile ? Quel est le résultat de l'exécution ?

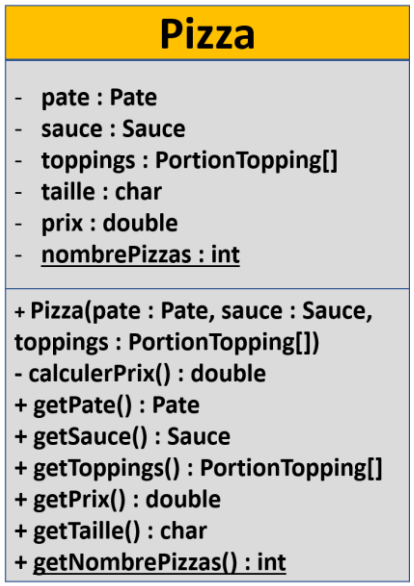
```
System.out.println(pizzaExotique.getPate().getQuantiteUtilisee());
System.out.println(pizzaKebab.getPate().getQuantiteUtilisee());
```

Exercice IV : plus d'attributs statiques

Dans cet exercice nous allons essayer d'ajouter des rabats de prix pour les pizzas et de calculer quelques statistiques concernant leur production.

- Nous allons ajouter un attribut statique nombrePizzas de type int dans la classe Pizza. Cet attribut sera initialisé à zéro. A chaque fois qu'une pizza est produite, on augmente cet attribut par 1. Pour chaque 10eme pizza, on fait un rabat de prix de 10% sur le coût de la pizza.
    - Ecrivez à nouveau votre méthode calculerPrix() de la classe Pizza pour prendre cela en compte.
    - Ecrivez à nouveau votre constructeur de cette classe pour faire augmenter le nombre de pizzas à chaque instanciation

  Le diagramme de cette classe est maintenant :



  Un attribut/méthode statique est toujours souligné dans un diagramme de classe !

- Ajoutez un attribut statique nombreToppings de type int dans la classe PortionTopping, qui augment par 1 à la création de chaque topping. Dans la méthode main de la classe TD3 ecrivez du code qui permet de savoir combien de toppings on a en moyenne pour chaque pizza.