EXAMEN FINAL PHP POO

UTILISER TOUTES VOS CONNAISSANCES ET DE MANIÈRE LOGIQUE ET ADÉQUATE POUR PARVENIR AU RESULTAT ATTENDU.

Le but de ce TP noté est d'écrire quelques classes simples permettant de vérifier que vous maitrisez la mise en œuvre des mécanismes de base pour la création de classes et d'objets, de composition, de délégation et de responsabilité, de sous-typage par héritage ou implémentation d'interfaces, cycle de vie des objets, ainsi que l'utilisation de collections élémentaires pour la réalisation de petits algorithmes simples en PHP oo.

On souhaite écrire des classes modélisant des articles (en vente dans un supermarché par exemple), ainsi qu'un panier d'articles permettant de les stocker.

Architecture MVC

Exercice 1 - Des articles (3 points)

1. Écrire une classe Item représentant un article ayant deux champs, Name pour son nom de type String et Price pour son prix en centimes de type long (on supposera que toutes les manipulations de prix doivent se faire avec des entiers long). Cette classe devra posséder deux accesseurs permettant d'obtenir le nom de l'article et son prix, ainsi qu'un constructeur qui permet d'en créer des instances. On veut que le prix d'un article et son nom ne changent pas pour toute la durée de vie de l'objet.
   Écrire une classe Main permettant de tester la création d'un article et le fonctionnement des accesseurs de la manière suivante:

2. Item = new Item("corn flakes", 500);
3. Var-dump(item.getPrice()); // affiche: 500
4. Var-dump(item.getName()); // affiche: corn flakes
5. On souhaite que lorsque l'on affiche un article avec Var-dump (), le nom de l'article s'affiche suivi de deux points (":"), d'un espace et de son prix en euros, où les centimes sont précédés d'un point ('.'), le tout suivi d'un espace et du symbole '€'. Par exemple:
6. Item item = new Item("corn flakes", 500)
7. Var-dump (item); // affiche: corn flakes: 5.00 €
8. Item chewingGum = new Item("chewing gum",403);
9. Var-dump (chewingGum); // affiche: chewing gum: 4.03 €

Exercice 2 - Le panier d'achats (8 points)

1. Écrire une classe ShoppingCart modélisant un panier d'articles dans lequel on peut:
   - ajouter un article avec addItem();
   - retirer un article avec removeItem(), qui devra renvoyer false si l'item que l'on essaye de supprimer n'existe pas;
   - connaître le nombre d'articles avec itemCount();
   - calculer le prix total du panier avec totalPrice() (vous écrirez en commentaire au début de la méthode quel est l'ordre de grandeur de la complexité de cette méthode);
   -
2. Ajouter un poids (weight de type int) exprimé en grammes à la classe Item et modifiez le constructeur pour qu'il accepte ce paramètre.
   Modifier l'implantation de votre panier pour que l'on ne puisse pas ajouter d'article dans le panier si le poids de ce dernier doit dépasser 10 kg. Vous utiliserez pour cela une exception
   Attention: le test du poids devra se faire en temps constant (O(1)). Modifier
   Indication: penser que removeItem() a une valeur de retour.

3. On souhaite que chaque panier d'achat créé puisse disposer automatiquement à sa création d'un numéro de série unique (qui commence à 1 et qui est incrémenté de 1 à chaque nouveau panier créé), et qui soit connu comme l'identifiant (id) de ce panier.
   Ajoutez une méthode getId() à la classe ShoppingCart qui retourne cet entier de type int, et tout ce dont vous avez besoin pour l'implémenter

4. Ajouter à la classe ShoppingCart une méthode toString() qui retourne une représentation du contenu du panier, commençant par l'identifiant unique du panier et le nombre d'articles contenus, puis affichant tous les articles du panier, un article par ligne.

Exercice 3 - Des produits frais (3 points)

Certains articles sont particulièrement frais et peuvent nécessiter la prise en compte d'une date limite de consommation. Néanmoins, lorsqu'on les met dans un panier d'achat, ils se comportent comme des articles classiques et ont un nom, un prix et un poids.

1. Écrire une classe (FreshItem) qui correspond à un article frais ayant, en plus des informations stockées dans la classe Item, un champ bestBeforeDate de type String correspondant à la date limite de consommation au format YYYY-MM-DD.

L'affichage d'un article frais par var_dump() doit afficher les informations de l'article dans le même format que pour un Item, mais précédées de la date limite de consommation.

Exercice 4 - La petite note (6 points)

On souhaite maintenant éditer des factures, mais pas seulement d'articles ou d'articles frais, mais plus généralement pour une liste de choses qui peuvent être payées.

1. Pour commencer, on représente par le type Ticket des billets d'évènements ou de spectacles qui peuvent être vendus dans le même magasin que nos articles (Item) ou nos articles frais (FreshItem).
   Un billet est représenté par une référence (reference de type String) et par un prix en centimes d'euros (price de type long).
   Écrire une classe Ticket avec les champs nécessaires et un constructeur permettant de créer, par exemple:
2. Ticket ticket = new Ticket("RGBY17032012 - Walles-France", 9000);
3. On souhaite maintenant disposer d'un type Payable, qui dispose des méthodes:
   - label() qui retourne une String représentant une descritpion textuelle de ce qui doit être payé;
   - cost() qui retourne le coût de ce qui doit être payé en centimes (un entier long);
   - taxRatePerTenThousand() qui retourne la proportion du coût qui est de la taxe, exprimée en centièmes de pourcents (en "pour-dix-mille") sous la forme d'un entier long. Par exemple, dans cette unité, 550 représente un taux de taxe de 5,5% et 1960 représente un taux de taxe de 19,6%;

   Définir le type Payable et modifier la classe Ticket de sorte que le code suivant fonctionne (on considérera que toutes les instances de la classe Ticket sont par défaut taxées à 25%):

4. On représente maintenant une facture comme une liste de choses à payer. Créer une classe Invoice qui dispose pour l'instant d'un seul constructeur sans argument et d'une unique méthode add(Payable p) qui ajoute à la liste de choses à payer l'argument p.
5. Toutes les instances de la classe Item sont taxées à 10% par défaut.
6. Pour les articles frais, instances de FreshItem, la taxe est réduite de 0,1% par tranche d'un kilo de produit. Par exemple, les sardines en boite sont taxées à 10%, mais les sardines fraiches sont taxées à 10% s'il y en a moins d'1 kg, et à 10% - 0,1% s'il y en a entre 1 et 2 kg...
7.
8. On souhaite ajouter dans la classe Invoice deux méthodes: totalAmount() qui retourne un long représentant le montant total des choses à payer de cette facture,

et totalTax() qui retourne un long représentant le montant total des taxes de cette facture.

9. Evidement chaque factures doit être enregistrer en base de donner (produits,qty,total ,tax)

Bonus :

1- Illustrer cela en utilisant le Template de rendu (twig) installer via le gestionnaire de package composer

Afin d'avoir un visuel de votre supermarché.

2- Utiliser une librairie afin de pouvoir télécharger la facture au format PDF.

EXEMPLE : HTMLTOPDF

https://www.html2pdf.fr/