TEK-UP
Ecole Supérieure Privée
Technologie & Ingénierie

Ecole Supérieure Privée Technologies & Ingénierie

TD2-Design Pattern (Structure)

Exercice 1 :

A un distributeur, les clients ont le choix entre 3 types de café : Colombia (1 D),Expresso (1.2 D) et Deca (0.8D). Ils peuvent ajouter les suppléments suivants : Lait, Sucre, Caramel, Chantilly. L'ajout de sucre ou de lait est facturé 0.2 D, de caramel 0.4D et de chantilly 0.8D. Il faut garder à l'esprit que l'ajout futur de nouveaux ingrédients (café ou supplément) doit être simplifié. On veut afficher dans la console le prix du café choisi avec ses suppléments.

1) Quel est le patron de conception qui peut être utilisé pour ce distributeur
2) Donner le diagramme de classe modélisant ce distributeur
3) préciser l'implémentation des méthodes, et donner le code pour afficher le prix d'un Expresso avec sucre, et d'un Colombia Chantilly Caramel.

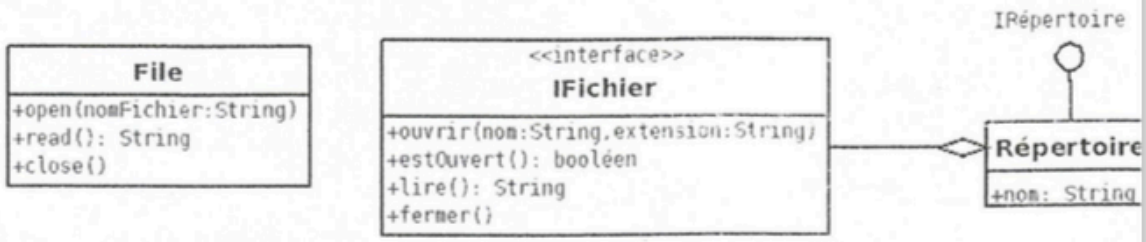
Exercice 2 :

Dans le cas d'un logiciel qui manipule le style et le format de police dans un texte, le trait a une épaisseur (simple ou double), une continuité (continu, en pointillé), une ombre ou pas, des coordonnées. Les caractéristiques d'épaisseur, de continuité et d'ombre sont des attributs intrinsèques à un trait, tandis que les coordonnées sont des attributs extrinsèques. Plusieurs traits possèdent des épaisseurs, continuité et ombre similaires. Ces similitudes correspondent à des styles de trait. En externalisant les attributs intrinsèques des objets (style de trait), on peut avoir en mémoire une seule instance correspondant à un groupe de valeurs (simple - continu - sans ombre, double - pointillé - ombre).

1) Quel est le patron de conception qui peut être utilisé pour ce distributeur
2) Donner le diagramme de classe modélisant ce distributeur

Exercice 3 :

On dispose des classes présentées sur la figure suivante :



Question

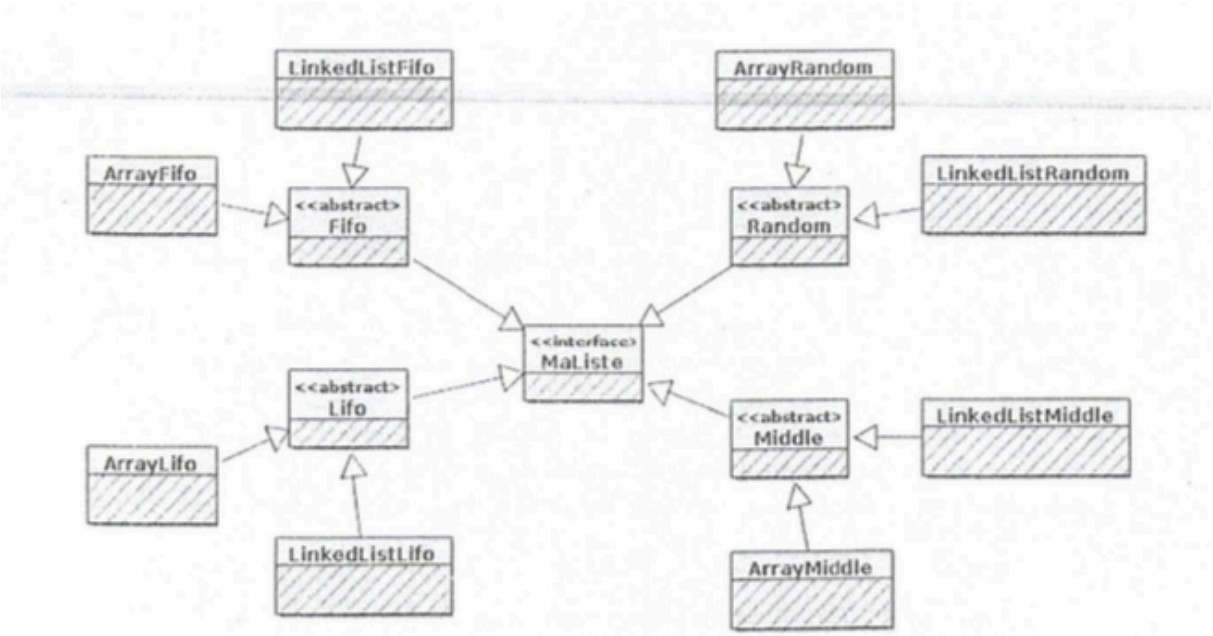
Une application est écrite en fonction de l'interface IRépertoire. On souhaiterait y intégrer une gestion de répertoires comprenant des fichiers manipulés via la classe File. Proposer une architecture permettant cela en maximisant l'utilisation du code déjà existant. On précisera comment peut être implémentée chaque nouvelle méthode introduite.

Exercice 4 :

On souhaite réaliser le projet suivant : créer une interface **MaListe** qui gère des ensembles d'entiers et qui contient trois fonctions : push, pop, et isEmpty.

Cette interface pourra être déclinée par des classes abstraites Lifo, Fifo, Random (qui affichent un nombre choisi au hasard dans l'ensemble quand on appelle pop) et Middle (qui affiche le nombre rangé au centre de la liste quand on appelle pop). De plus, chacune de ces listes pourra être implémentée à l'aide d'un tableau ou d'un système de liste chaînée.

Voici le schéma de classe du projet :



Après, vous vous rendez compte que, le jour où vous voudrez ajouter la gestion des ensembles avec des arbres binaires, il vous faudra rajouter 4 classes, et le jour où vous souhaiterez ajouter un système de gestion d'ensemble qui affiche le sort à chaque fois le plus petit élément de la liste, il vous faudra rajouter deux classes. En résumé, si vous avez N abstractions et M implémentations, le jour où vous souhaitez rajouter une seule implémentation, vous devrez écrire N classes, et le jour où vous souhaitez rajouter une seule abstraction, vous devrez rajouter M classes. Trouver un moyen pour améliorer ce modèle.