Ilyass Ikrimah
Mouazer Rayane

Compte rendu des TP

Choix des TP

Nous avons choisi ces TP car nous ne sommes pas parfaitement à l'aise et très forts en programmation et nous n'avons jamais fait de programmation orientée objet.Nous avons donc opté pour ce TP de niveau débutant.
Dans le TP1 nous devons créer différentes classes qui nous permettent de gérer une bibliothèque. Cette bibliothèque comporte des lecteurs des auteurs des emprunts et des livres qui peuvent être empruntés par des lecteurs .
Dans le TP2 nous devons créer différentes classes qui nous permettent de gérer un magasin qui comporte des produits , des clients et des commandes .
Pour ce qui est de la compilation nous avons réalisé pour chaque classe un fichier header .h et un fichier .cpp puis nous avons créé un makefile pour compiler tout.

TP1

Création des classes

Classe Date :

Cette classe "Date" a été créée pour stocker des informations sur une date (jour, mois, année). Elle contient les getters et setters demandés et nous avons ajouter une fonction qui permet de verifier si la date est valide en s'assurant que le jour est compris entre 1 et 31, le mois entre 1 et 12 et en gérant les cas spécifiques pour les mois de février et les mois à 30 jours, cette fonction s'appelle "isDate" . La fonction "assert" est utilisée pour vérifier que la date stockée est valide lors de sa création. Cette classe sera utilisée dans les classes Livre et Emprunt . on a donc inclu inclu le fichier header Date.h dans Livre.h et Emprunt.h

Classe Auteur :

Cette classe contient des membres privés : l'identifiant de l'auteur (_idn), le nom (_nom), le prénom (_prenom) et la date de naissance (_daten) de l'auteur. Elle contient des

constructeurs pour initialiser ces membres, ainsi que des getters et setters. Enfin, il y a une surcharge de l'opérateur << pour permettre l'affichage des informations de l'auteur.

Classe Livre :

La classe Livre a été créée pour représenter un livre avec les informations suivantes le titre, l'auteur, la langue, le genre, la date de publication, l'ISBN, la liste des emprunteurs et l'état. Elle utilise la classe Auteur, qui contient des informations sur l'auteur du livre. La classe Livre utilise également un vecteur de chaînes pour stocker la liste des emprunteurs du livre. Les fonctions de la classe permettent de récupérer ces informations, de les mettre à jour et d'afficher la liste des emprunteurs. la fonction ajouterunempruntid a été cree pour ajouter l'identifiant d'un emprunteur a la fin de la liste des emprunts. Il y a également une surcharge de l'opérateur << pour afficher les informations du livre.

Classe lecteur :

La classe Lecteur définit un objet représentant un lecteur d'une bibliothèque. Elle contient l'identifiant, le nom, le prénom et la liste des numéros ISBN des livres empruntés par le lecteur. Elle définit également des méthodes pour accéder et mettre à jour ces informations, ainsi qu'une méthode pour afficher la liste des livres empruntés et une méthode pour ajouter un numéro ISBN à la liste des livres empruntés. La classe inclut également un constructeur pour initialiser les membres lors de la création d'un objet Lecteur.

classe Emprunt :

La classe Emprunt contient 3 attributs: une date d'emprunt, un ISBN (numéro international normalisé du livre) et un identifiant de lecteur. Elle contient des méthodes pour accéder à ces attributs (getdateemprunt(), getISBN(), getidentifiant()) ainsi qu'une méthode pour mettre à jour les informations d'un emprunt (setemprunt()). Cette classe utilise également une classe Date qui semble être définie ailleurs.

Classe Bibliotheque :

Cette classe contient 4 attributs qui sont 4 listes de lecteurs, de livres, d'emprunts et d'auteurs. C'est ici qu'on a créé le plus de fonctions afin de gérer la bibliothèque comme la fonction pour emprunter un livre ou le restituer en veillant à ce qu' un lecteur qui a déjà emprunter un livre ne peut pas en emprunter le même une autre fois avant de le rendre ni par aucun autre lecteur, la fonction classement qui donne le lecteur qui dispose du plus grand nombre d'emprunts.

Problème et difficultées rencontrés :

- Pour la fonction classement on a pas pu faire un classement du premier 2eme et 3eme. On a juste pu donner le lecteur avec le plus d'emprunt mais pas e 2eme ou le 3eme

- Pour la fonction emprunter un livre on utilise la fonction ajouter_isnb_lecteur (qui fonctionne correctement si on la test directement sur un lecteur sauf que on l'ajoutant dans la fonction qui permet l'emprunt d'un livre elle ne rajoute pas l'isnb comme prévu à la fin de la liste des isbn empruntés du lecteur et nous n'avons pas résolu le problème

A part ça le reste fonctionne correctement les classes sont toutes créées sans problème les fonctions demandées fonctionnent toutes .

TP3

Création des classes

Classe Produit :

Dans cette classe **"Product"** , il y a des **constructeurs** qui permettent de créer des objets de la classe avec des valeurs par défaut ou en utilisant des paramètres spécifiques tels que le titre, la description, la quantité et le prix. Il y a aussi des méthodes **"getters" et "setters"** pour accéder et modifier les attributs privés de la classe qui sont **_titre, _description, _quantité et _prix**. De plus, il y a **une surcharge de l'opérateur <<** pour afficher les informations de l'objet de manière plus lisible.

Classe Client :

Dans cette classe **"Client"**, il y a des constructeurs qui permettent de créer des objets de la classe avec des valeurs par défaut ou en utilisant des paramètres spécifiques tels que le prénom, le nom, le panier (vector<product>) et l'ID. Il y a aussi des méthodes "getters" pour accéder aux attributs privés de la classe qui sont **_prenom, _nom, _id et _panier.** Il y a aussi des **méthodes** pour **ajouter des produits au panier, vider le panier, modifier la quantité d'un produit et supprimer un produit du panier.** et enfin la classe comprend une surcharge de **l'opérateur << pour** afficher les attributs de l'objet .

Classe Commande :

Le code ci-dessus définit une classe appelée **"Commande"** qui est définie dans le namespace "commande". Cette classe possède un constructeur qui prend en entrée un **objet "Client", un booléen "confirmation", un vector de produits "products" et un entier "id".** Les déclarations des setters permettent de définir les valeurs des membres privés de la classe, à savoir "client", "confirmation", "products" et "id". Les déclarations des getters permettent de récupérer les valeurs des membres privés de la classe. Il y a également une **surcharge d'opérateur "<<" pour afficher l'objet de la classe Commande.**

Classe Magasin :

On définit dans ce code une classe appelée "Magasin" dans l'espace de noms "magasin". La classe possède un constructeur qui prend des références à **des vecteurs d'objets de type "Produit", "Client" et "Commande**" et initialise les variables membres **privées _produits, _clients et _commandes** respectivement.

La classe possède aussi des **méthodes publiques** pour ajouter et récupérer des produits, des clients et des commandes, telles que **"addproduct", "getproducts", "addclient"** et **"getclients".** Elle possède également des méthodes pour afficher tous les clients ou produits (**showclients() , showproducts()**) ou de rechercher un client ou un produit à l'aide de différents arguments .

Enfin on a implémenté des **méthodes publiques** pour ajouter , supprimer ou modifier la quantité d' un produit **dans un panier d'un client .**

Problème et difficultées rencontrés :

- Pour la fonction commande::Commande::getclient() on a eu des problème avec l'accès des attributs du client , alors on a ajouté * pour le différencier

- Pour la fonction magasin.confirmcommande() le compilateur ne montre aucune erreur mais la fonction n'arrive pas a changer la valeur de confirmation .

```
std::cout <<"confirmation avant " << c1.getconfirmation()<<std::endl;
IKRIMAHSTORE.confirmcommande(1);
std::cout <<"confirmation apres "  << c1.getconfirmation()<<std::endl;
```

```
confirmation avant 0
confirmation apres 0
```

Méthode de travail en groupe :

On a travaillé chacun de notre côté rayan a plus travaillé sur le TP1 et ilyass plus sur le TP3 mais on se retrouvait souvent pour réviser ensemble et mettre en commun nos travaux et essayer de résoudre ensemble les problèmes de chacun.

Compilation :

Au fur et à mesure de l'avancement des tp et on a corrigé les erreurs puis au final nous avons realiser un makefile pour compiler le tout facilement et efficacement.

Logiciels utilisés :

Nous avons utilisé le logiciel code blocks pour la réalisation de nos programmes. En plus pour le choix de programmation ou nous avons utilisé tout notre connaissance de cours et nous avons utilisé la logique pour arriver à programmer toutes les classes nécessaires avec tous les attributs et les fonctions pour assurer le bon fonctionnement et en cas de blocage ou d'erreurs nous avons consulté plusieurs sites internet pour arriver à bien résoudre tous les problèmes rencontrés.