PROJET C++

Gestion de Bibliothèques

NOTICE

Alan Mary

Table des matières

1. Introduction

Ce projet développe une application C++ de gestion de bibliothèques, permettant de gérer les livres, adhérents, emprunts et prêts inter-bibliothèques. Le projet inclu: héritage, polymorphisme, templates, listes chaînées et gestion mémoire.

2. Architecture du projet

2.1 Organisation des fichiers

Le projet respecte la convention de 2 fichiers par classe (.h et .cpp), plus main.cpp.

Fichier	Description
Noeud.h	Template de nœud pour liste chaînée
Liste.h	Template de liste chaînée générique
Livre.h / Livre.cpp	Classe de base pour tous les livres
Roman.h / Roman.cpp	Classe dérivée pour les romans
BD.h / BD.cpp	Classe dérivée pour les bandes dessinées
Theatre.h / Theatre.cpp	Classe dérivée pour les pièces de théâtre
Recueil.h / Recueil.cpp	Classe dérivée pour les recueils de poésie
Album.h / Album.cpp	Classe dérivée pour les albums
Biblio.h / Biblio.cpp	Classe représentant une bibliothèque
Adherent.h / Adherent.cpp	Classe représentant un adhérent
main.cpp	Programme principal avec tests

2.2 Relations entre les classes

- **Héritage** : Roman, BD, Theatre, Recueil et Album héritent de Livre
- **Composition** : Biblio contient une Liste<Livre*> (ses livres)
- **Composition** : Adherent contient une Liste<Livre*> (ses emprunts)
- **Association** : Adherent est lié à une Biblio* (sa bibliothèque d'inscription)
- **Association** : Livre est lié à une Biblio* (son propriétaire)

3. Description des classes

3.1 Classes templates : Noeud<T> et Liste<T>

3.1.1 Classe Noeud<T>

Représente un nœud générique dans une liste chaînée.

Attribut	Type	Description
info	T	Donnée stockée dans le nœud
suivant	Noeud<T>*	Pointeur vers le nœud suivant

Méthode	Description
Noeud() / Noeud(T)	Constructeurs
getInfo() / setInfo()	Accesseurs pour la donnée
getSuivant() / setSuivant()	Accesseurs pour le lien
operator=	Opérateur d'affectation

3.1.2 Classe Liste<T>

Implémente une liste simplement chaînée générique.

Attribut	Type	Description
tete	Noeud<T>*	Pointeur vers le premier nœud
taille	int	Nombre d'éléments

Méthode	Description
insererDebut() / insererFin()	Insertion en tête ou en queue
inserer(T, int)	Insertion à une position donnée
supprimerDebut() / supprimerFin()	Suppression aux extrémités
supprimer(T)	Suppression par valeur
rechercher(T)	Recherche d'un élément
contient(T)	Vérifie la présence d'un élément
vider()	Supprime tous les éléments
operator+ / operator=	Concaténation et affectation

3.2 Classe Livre et ses classes dérivées

3.2.1 Classe Livre (classe de base)

Représente un livre générique avec ses caractéristiques communes.

Attribut	Type	Description
code	string	Identifiant unique du livre
auteur	string	Nom de l'auteur
titre	string	Titre de l'ouvrage
editeur	string	Maison d'édition
isbn	string	Numéro ISBN
etat	Etat (enum)	libre, emprunte ou prete
audience	Public (enum)	ados, jeunesse, tout_public, adulte
proprietaire	Biblio*	Bibliothèque propriétaire

Méthode	Description
afficher()	Affiche les informations du livre
emprunter()	Passe l'état à emprunté si libre
rendre()	Remet l'état à libre
estdispo_emprunt(Biblio*)	Vérifie la disponibilité pour emprunt

3.2.2 Classes dérivées de Livre

Classe	Attribut spécifique	Description
Roman	genre : Genre (enum)	Type de roman (littérature, noir, etc.)
BD	dessinateur : string	Nom du dessinateur
Theatre	siecle : int	Siècle de la pièce
Recueil	vers, prose : bool	Type de poésie
Album	dessin, photo : bool	Type d'illustrations

Chaque classe dérivée redéfinit la méthode `afficher()` pour inclure ses attributs spécifiques (polymorphisme).

3.3 Classe Biblio

Représente une bibliothèque du réseau.

Attribut	Type	Description
nom	string	Nom de la bibliothèque
adresse	string	Adresse physique
code	string	Code identifiant unique
liste	Liste<Livre*>	Collection de livres

Méthode	Description
ajouterLivre(Livre*)	Ajoute un livre à la collection
supprimerLivre(Livre*)	Supprime un livre (avec delete)
chercherLivre(code)	Recherche par code
chercherLivre_IBSN(isbn)	Recherche par ISBN
acheterLivre(Livre*)	Achète et devient propriétaire
preterLivre(code, Biblio*)	Prête un livre à une autre bibliothèque
demanderLivre(isbn, Biblio*)	Demande un livre à une autre bibliothèque
rendreLivre(code)	Rend un livre emprunté à son propriétaire

Le destructeur libère la mémoire des livres dont la bibliothèque est propriétaire.

3.4 Classe Adherent

Représente un adhérent inscrit à une bibliothèque.

Attribut	Type	Description
nom, prenom	string	Identité de l'adhérent
adresse	string	Adresse postale
numAdherent	int	Numéro d'adhérent unique
biblio	Biblio*	Bibliothèque d'inscription
livresEmpruntes	Liste<Livre*>	Liste des livres empruntés
nbMaxEmprunt	int	Limite d'emprunts simultanés
nbAdherents	static int	Compteur de tous les adhérents

Méthode	Description
peutEmprunter()	Vérifie si la limite n'est pas atteinte
emprunterLivre(code)	Emprunte un livre par son code
rendreLivre(code)	Rend un livre emprunté
afficher()	Affiche les informations de l'adhérent
afficherLivresEmpruntes()	Liste les livres actuellement empruntés
getNbAdherents() [static]	Retourne le nombre total d'adhérents

4. Notions C++ utilisées

Notion	Utilisation dans le projet
Héritage	Livre → Roman, BD, Theatre, Recueil, Album
Polymorphisme	Méthode afficher() redéfinie dans chaque classe dérivée
Templates	Classes Noeud<T> et Liste<T> génériques
Listes chaînées	Implémentation complète avec insertion, suppression, recherche
Membres statiques	nbAdherents dans Adherent pour compter les instances
Surcharge d'opérateurs	operator= et operator+ dans Liste<T>
Énumérations	Etat, Public, Genre pour typer les valeurs
Constructeur de copie	Implémenté dans toutes les classes
Gestion mémoire	new/delete, destructeur de Biblio libère ses livres
Encapsulation	Attributs private, accès via getters/setters
Déclaration anticipée	class Biblio; dans Livre.h (inclusion circulaire)

5. Fonctionnalités implémentées

5.1 Gestion des livres

- Création de livres de différents types (Roman, BD, Théâtre, Poésie, Album)
- Achat de livres par une bibliothèque (devient propriétaire)
- Recherche de livres par code ou par ISBN
- Suppression de livres (perte ou mise au pilon)

5.2 Gestion des emprunts

- Emprunt d'un livre par un adhérent (vérification de disponibilité)
- Retour d'un livre emprunté
- Respect de la limite d'emprunts par adhérent
- Impossibilité d'emprunter un livre déjà emprunté ou prêté

5.3 Prêts inter-bibliothèques

- Une bibliothèque peut prêter un livre à une autre
- Le propriétaire du livre reste la bibliothèque d'origine
- La bibliothèque emprunteuse peut retourner le livre
- Seuls les livres libres peuvent être prêtés entre bibliothèques

6. Tests réalisés

Le fichier main.cpp contient une suite complète de tests :

Test	**Description**
Test 1	Création des bibliothèques
Test 2	Création et achat de livres de différents types
Test 3	Recherche de livres par code et par ISBN
Test 4	Création des adhérents et association à leur bibliothèque
Test 5	Emprunt de livres par les adhérents
Test 6	Vérification de la limite d'emprunt
Test 7	Retour de livres empruntés
Test 8	Prêt inter-bibliothèques
Test 9	Comportement avec un livre prêté
Test 10	Retour d'un prêt inter-bibliothèques
Test 11	Affichage des informations
Test 12	Test du polymorphisme

Tous les tests s'exécutent correctement et le programme arrive à son terme sans erreur.