Université Abdelmalek Essaâdi
Faculté des Sciences et Techniques – Tanger
Année Universitaire : 2025/2026
Module : POO en C++, Licence IDAI
Prof. Ikram BEN ABDEL OUAHAB

ATELIER 4

EXERCICE 1 : GESTION D'UNE MÉDIATHÈQUE

On souhaite concevoir une application C++ pour gérer une **médiathèque** contenant plusieurs types de ressources : **Livres**, **Magazines** et **Vidéos**.

1. Créer une classe de base Ressource avec des attributs communs (id, titre, auteur, annee) et des méthodes afficherInfos() et telecharger().
2. Créer les classes dérivées **Livre**, **Magazine** et **Video** héritant de ***Ressource***, chacune ajoutant un attribut spécifique et redéfinissant ***afficherInfos***() et ***telecharger***().
3. Créer une classe **Telechargeable** contenant une méthode ***telecharger***() et ***afficherMessage***(). Les classes **Livre**, **Magazine** et **Video** héritent également de *Telechargeable* (**héritage multiple**).
4. Créer une classe **Mediatheque** permettant d'ajouter, d'afficher et de rechercher des ressources. Implémenter la **surcharge** de la méthode rechercher() (par titre, année, auteur + année).
5. Dans **Ressource**, surcharger l'opérateur == pour comparer deux ressources selon leur identifiant.
6. Dans ***main***(), créer plusieurs objets, tester les méthodes afficherInfos(), telecharger() (avec qualification de classes en cas de conflit), et comparer deux ressources.

EXERCICE 2: SYSTÈME DE GESTION BANCAIRE

Une banque souhaite développer une application en C++ pour gérer les **comptes de ses clients** et permettre à certains **agents autorisés** d'effectuer des opérations confidentielles. Chaque compte est associé à un client et contient des informations sensibles (numéro de compte, solde, code secret).

1. Créer une classe Client contenant les informations de base : nom, CIN et un identifiant client.
2. Créer une classe CompteBancaire représentant un compte, avec :
   - un numéro de compte,
   - un solde,
   - un code secret (non accessible publiquement),
   - et une référence vers le client propriétaire.
3. Créer une classe AgentBancaire qui représente un employé autorisé à effectuer des opérations confidentielles sur les comptes (consultation du code secret, transfert entre comptes, etc.). Les méthodes d'un agent doivent pouvoir accéder à certaines informations privées des comptes et des clients **sans tout exposer au reste du programme**.
4. Ajouter une classe Banque pour centraliser la gestion des clients et des comptes (ajout, affichage, transfert, etc.). Cette classe doit pouvoir afficher les détails internes d'un compte pour des raisons d'audit interne.

Contraintes :

- Aucune donnée sensible (comme le code secret) ne doit être directement accessible depuis l'extérieur.
- Seules certaines classes ou fonctions internes doivent y avoir accès (à l'étudiant de concevoir la bonne relation entre elles).
- L'accès à certaines informations devra être possible **sans recourir à des accesseurs classiques (get/set)**.

Dans main() :

- Créer plusieurs clients et comptes.
- Simuler des dépôts, retraits, et transferts entre comptes.
- Afficher un rapport d'audit via la classe Banque montrant les soldes et informations confidentielles.