INF1010

Programmation Orientée-Objet

Travail pratique #3
Héritage

Objectifs :	Permettre à l'étudiant de se familiariser avec le concept d'héritage
Remise du travail :	Dimanche 21 février 2016, 23h
Références :	Notes de cours sur Moodle & Chapitre 8 du livre Big C++ 2e éd.
Documents à remettre :	Les fichiers .cpp et .h complétés réunis sous la forme d'une archive au format .zip.
Directives :	Directives de remise des Travaux pratiques sur Moodle Les en-têtes (fichiers, fonctions) et les commentaires sont obligatoires. Les travaux dirigés s'effectuent obligatoirement en équipe de deux personnes faisant partie du même groupe. Veuillez suivre le guide de codage

Travail à réaliser

Dans ce TP nous poursuivrons l'implémentation de notre bibliothèque en y incorporant de nouveaux éléments.

Lors de l'implémentation de vos différentes classes dans un projet, vous pouvez remarquer que certaines d'entre elles partagent des points communs, elles s'apparentent à une même catégorie à travers leurs comportements ou/et leurs spécifications communes. Dans la programmation orienté objet, cela est représenté par une notion très importante qui est **l'héritage**.

L'héritage permet à partir d'une classe de base de créer des classes dérivées qui auront un comportement semblable et ainsi regrouper dans une même famille d'objets similaires.

Votre bibliothèque a eu la chance de recevoir une grosse subvention pour la nouvelle année, elle a pu acquérir grâce à cela du contenu multimédia. Par conséquent, elle propose des offres pour certaines catégories de personnes (les étudiants et les professeurs). Dans ce TP, vous allez mettre en œuvre les notions d'héritage afin d'enrichir le système développé lors des précédents TPs. Plusieurs classes sont reprises, et nous vous demandons rajouter de nouvelles classes.

ATTENTION : Tout au long du TP, assurez-vous d'utiliser les opérateurs sur les objets et non sur leurs pointeurs ! Vous devez donc déréférencer les pointeurs si nécessaire.

ATTENTION : C'est à vous de choisir la portée de vos attributs (private, protected ou public) sauf si explicitement mentionné.

Classe *Abonne*

Créer une classe *Abonne* qui représente un abonné de la bibliothèque et qui va servir de classe de base. La classe Abonne du TP précédent est reprise, mais vous devez apporter les modifications suivantes :

Cette classe contient les nouveaux attributs :

- *LimiteEmprunts*(entier), la limite d'emprunt pour un abonne classique est de 2. Cet attribut doit être déclaré comme *private*

Les méthodes suivantes doivent être implémentées ou modifiées :

- Une méthode pour obtenir le nombre d'emprunts limite
- Une redéfinition de la méthode *estEmprunte()* qui prend en paramètre une *cote* et vérifie si l'abonné possède un emprunt associé à cet objet. Si oui, la méthode retourne *true*, sinon *false*

- Une redéfinition de la méthode *retirerEmprunt ()* qui prend en paramètre un *ObjetEmpruntable.* La méthode enlève l'emprunt associé à cet objet du vecteur seulement s'il est présent. La méthode retourne un booléen *true* si l'emprunt est retiré et *false* sinon

Classe *Etudiant*

Cette classe dérive de la classe Abonne.

Cette classe possède les attributs supplémentaires suivants :

- Ecole (string), qui correspond à l'école de provenance de l'étudiant

Les méthodes suivantes doivent être implémentées :

- Un constructeur par défaut qui initialise en plus l'attribut Ecole à la chaîne vide
- Le destructeur pour cette classe
- Un constructeur par paramètres qui initialise les attributs selon les paramètres
- Les méthodes d'accès et de modification du nouvel attribut
- Une redéfinition de la méthode pour obtenir le nombre d'emprunts, en tenant compte du fait qu'un étudiant à une limite d'emprunt 2 fois plus élevée qu'un abonné classique
- L'opérateur <<, qui affiche les informations de l'étudiant, tel que présenté dans l'exemple à la fin du document. Il doit utiliser l'opérateur << de la classe de base

Classe *EtudiantBaccalaureat*

Cette classe dérive de la classe Etudiant et représente un étudiant au baccalauréat de l'école Polytechnique Montréal.

Les méthodes suivantes doivent être implémentées :

- Un constructeur prenant en paramètre un matricule, un nom, un prénom et un âge et qui initialise l'attribut Ecole à « Polytechnique Montreal »
- Une redéfinition de la méthode pour obtenir le nombre d'emprunts, en tenant compte du fait qu'un étudiant au baccalauréat à une limite d'emprunts 2 fois plus élevé qu'un étudiant classique

Classe *Professeur*

Cette classe dérive de la classe Abonne.

Cette classe possède les attributs supplémentaires suivants :

- Un vecteur de string qui correspond aux différentes écoles où il enseigne.

Les méthodes suivantes doivent être implémentées :

- Un constructeur par défaut
- Le destructeur pour cette classe
- Un constructeur par paramètres qui initialise les attributs en fonction des paramètres.
- La méthode d'accès au nouvel attribut
- Une méthode *ajouterEcole*() qui prend en paramètre un *string*, et l'ajoute au vecteur
- Une méthode *retirerEcole* () qui prend en paramètre un *string*. La méthode enlève l'école associée à ce string du vecteur. La méthode retourne un booléen true si l'école est retirée et false sinon.
- La redéfinition de la méthode obtenir *limiteEmprunts*() en tenant compte du fait qu'un enseignant peut emprunter un nombre limité de livre définie comme suit *limiteEmprunt = nombre Ecole x limite Emprunt Classique*
- L'opérateur <<, qui affiche les informations du Professeur, tel que présenté dans l'exemple à la fin du document. Il doit appeler l'opérateur << de la classe *Abonne*.

Classe Objet*Empruntable*

Cette classe va servir de classe de base aux différents éléments qui peuvent être empruntés. Cette classe vous est fournie. Vous n'avez pas besoin de la modifier.

Classe *Livre*

Créer une classe *Livre* dérivée de la classe *ObjetEmpruntable* et qui va représenter un livre dans la bibliothèque. Vous devez implémenter une nouvelle classe Livre sans tenir compte de celle du TP2.

Cette classe possède les attributs suivants :

- Auteur(string)
- Genre(string), qui représente la catégorie à laquelle appartient le livre

Livre implémente les méthodes suivantes :

- Un constructeur par paramètres qui initialise en plus les nouveaux attributs
- Des méthodes d'accès et de modifications pour les nouveaux attributs

- Une redéfinition de la méthode *rechercher*() qui va aussi vérifier si la chaîne de caractères est présente dans les nouveaux attributs de type string
- L'opérateur << qui affiche les informations du *Livre*, tel que présenté dans l'exemple à la fin du document. Il doit appeler l'opérateur << de la classe *ObjetEmpruntable*

Classe *DVD*

Créer une classe *DVD* dérivée de la classe ObjetEmpruntable et qui représente un DVD dans la bibliothèque.

Cette classe possède les attributs suivants :

- Realisateur(string)
- Un vecteur de string qui correspond aux acteurs

DVD implémente les méthodes suivantes :

- Un constructeur par paramètre qui initialise les nouveaux attributs
- Des méthodes d'accès et de modifications pour les nouveaux attributs
- Une redéfinition de la méthode *rechercher*() qui vérifie si la chaîne de caractères est présente dans les nouveaux attributs de type string
- L'opérateur << qui affiche les informations du *Dvd*, tel que présenté dans l'exemple à la fin du document. Il doit appeler l'opérateur << de la classe *ObjetEmpruntable*

Classe *Emprunt*

La classe *Emprunt* sert à représenter l'emprunt d'un objet par un abonné.

Les attributs et méthodes liés au TP précédent restent inchangés, sauf si spécifié.

Cette classe va contenir les nouveaux attributs suivants :

- Pointeur de *DVD*

Emprunt implémente les méthodes suivantes :

- Un constructeur par paramètre pour chaque catégorie d'objet empruntable pour un abonné, elle va initialiser les autres types d'objets empruntables au pointeur *nullptr*
- Les méthodes d'accès et de modification des nouveaux attributs.
- Une redéfinition de la méthode *obtenirLivre*() en *obtenirObjetEmpruntable*() qui va renvoyer le pointeur non nul s'il y a, le pointeur nul sinon
- Un opérateur <<, qui affiche le matricule de l'abonné, puis appelle l'opérateur << de l'objet empruntable approprié ; tel que présenté dans l'exemple à la fin du document.

Classe *Bibliotheque*

La classe *Bibliotheque* est celle qui fait le lien entre toutes les classes précédentes.

Cette classe ressemble à ce qui a été fait dans le TP2

Bibliotheque contient les attributs suivants :

- Vecteur de pointeurs *d'abonnés*.
- Vecteur de pointeurs de *livres*.
- Vecteur de pointeurs de *DVD*.
- Vecteur de pointeurs d'*emprunt*.

Bibliotheque implémente les méthodes suivantes :

Indication : Certaines méthodes demandées sont très semblables à celles présentes dans le TP2

- Un constructeur par défaut.
- Un destructeur.
- Une méthode *trouveAbonne()* qui prend un matricule (string) en paramètre, recherche dans le vecteur d'abonnés et retourne un pointeur *Abonne* associé à ce matricule. Si aucun abonné n'est retrouvé, un pointeur nul est retourné.
- Une méthode *trouverObjetEmpruntable()* qui prend une cote (string) en paramètre, recherche l'objet empruntable et retourne un pointeur de type *ObjetEmpruntable*. Si aucun objet empruntable n'est retrouvé, un pointeur nul est retourné sinon.
- Une méthode *obtenirClasseObjet*() qui prend une cote (string) en paramètre, et renvoie le nom de la classe correspondant à l'objet ayant cette cote (« *Livre* », « *DVD* »).
- Une méthode *ajouterAbonne()* qui permet d'ajouter l'abonné reçu en paramètre, seulement s'il n'est pas déjà dans le vecteur. Si l'ajout a été fait, la méthode renvoie true, false sinon.
- La méthode *retirerAbonne()* qui permet de retirer l'abonné en utilisant le matricule reçu en paramètre. Avant de le retirer, il faut **retourner tous les éléments que l'abonné a emprunté** en appelant la méthode appropriée. Si le retrait a été fait, la méthode renvoie true, false sinon.
- La méthode *ajouterLivre()* qui permet d'ajouter le Livre reçu en paramètre seulement s'il n'est pas déjà dans le vecteur. Si l'ajout a été fait, la méthode renvoie true, false sinon.
- La méthode *ajouterDVD()* qui permet d'ajouter le *DVD* reçu en paramètre seulement s'il n'est pas déjà dans le vecteur. Si l'ajout a été fait, la méthode renvoie true, false sinon.
- La méthode *retirerObjetEmpruntable()* qui permet de retirer l'objet empruntable en utilisant la cote reçue en paramètre. Il est retiré seulement s'il n'est pas emprunté. La méthode retourne donc un booléen *true* s'il est retiré, sinon *false*.
- La méthode *rechercher()* qui prend en paramètre une chaîne de caractères (string),
    - Cette méthode va rechercher les différents éléments empruntables de la bibliothèque qui contiennent l'information désirée en **utilisant la méthode appropriée de la classe**. Pour chaque élément trouvé les informations seront affichées à l'aide de l'opérateur << de la classe *appropriée*.

    - N'oubliez pas de parcourir tous les éléments qui peuvent être empruntés
    - Si aucun rien n'a été trouvé, affichez un message.
- Une méthode *estEmpruntable()* qui prend en paramètre un *matricule* et un *objet empruntable.*
    - Cette méthode vérifie s'il est possible pour l'abonné d'effectuer l'emprunt en question. Il est possible pour un abonné d'emprunter un objet empruntable si ce dernier est disponible, qu'il a l'âge minimal requis, qu'il n'a pas déjà emprunté celui-ci et que sa limite d'emprunts n'est pas encore atteinte
    - Cette méthode doit retourner une valeur booléenne indiquant si l'emprunt est possible ou non.
- La méthode *emprunter()* qui prend en paramètres le matricule d'un abonné, la cote d'un objet empruntable et la date de retour associée en fonction du statut de l'abonné. Cette méthode doit retourner une valeur booléenne indiquant si l'emprunt a été fait ou non.
    - Elle doit s'assurer que le nombre d'emprunts ne dépasse pas la limite par type d'abonné.
    - Elle doit diminuer le nombre d'objets associés disponibles.
    - Indice : Utilisez la méthode *obtenirClasseObjet* pour prendre en compte la bonne classe de l'objet emprunté
- La méthode *retourner()* qui prend en paramètres le matricule d'un abonné et la cote d'un objet empruntable. Si l'abonné avait bien emprunté cet élément, l'emprunt en question est retiré du vecteur d'emprunts.
    - Cette méthode doit retourner une valeur booléenne indiquant si le retour a été fait ou non.
    - Elle doit augmenter le nombre d'objets empruntables disponibles.
- La méthode *infoAbonne()* qui prend en paramètre un matricule d'abonné et affiche les informations qui le concerne en utilisant l'opérateur << approprié.
- Un opérateur >> qui permet d'entrer un mot-clé à chercher, puis qui appelle la méthode *rechercher()*.
- L'opérateur += qui est défini 3 fois, avec un paramètre d'entrée différent :
    - Une définition pour chaque type d'abonné qui prend en paramètre un pointeur *de* la classe correspondante. Son comportement est similaire à *ajouterAbonne()*.
    - L'autre définition prend en paramètre un pointeur *Livre*. Son comportement est similaire à *ajouterLivre()*.
    - L'autre définition prend en paramètre un pointeur *DVD*. Son comportement est similaire à *ajouterDVD()*.
- L'opérateur -= qui est défini 2 fois, avec un paramètre d'entrée différent
    - Une définition prend en paramètre un pointeur *Abonne*. Son comportement est similaire à *retirerAbonne()*.
    - L'autre définition prend en paramètre un pointeur *ObjetEmpruntable*. Son comportement est similaire à *retirerObjetEmpruntable()*.

Main.cpp

Le programme principal contient des directives à suivre pour tester les différentes méthodes que vous devez implémenter dans les différentes classes. Vous n'avez pas à compléter le main, assurez-vous juste du bon fonctionnement des méthodes appelées.

Le résultat final devrait être similaire à ce qui suit :

```
CREATION ET AFFICHAGE DES ABONNES

John, Doe. 23 ans. #1839456
LISTE DE LIVRES :
Nicolas, Gagnon. 8 ans. #1630236
LISTE DE LIVRES :
Martin, Tremblay. 18 ans. #1269348
LISTE DE LIVRES :
Harry, Potter. 21 ans. #1782965
LISTE DE LIVRES :
Ecole de provenance : Pourdlard; Limite d'emprunts : 4
Hermione, Granger. 20 ans. #1876458
LISTE DE LIVRES :
Ecole de provenance : Pourdlard; Limite d'emprunts : 4
Tony, Stark. 42 ans. #1865487
LISTE DE LIVRES :
Ecole de provenance : Polytechnique Montreal; Limite d'emprunts : 4
Albus, Dumbledore. 78 ans. #p878546
LISTE DE LIVRES :
Limite d'emprunts : 6
LISTE DES ECOLES : Poudlard; UdM; Polytechnique;
_____
CREATION ET AFFICHAGE DES OBJETS EMPRUNTABLES

Information sur le livre :
GA403. Big C++. 2009. 8 ans et plus. Auteur : Cay HORTSTMANN; Genre : Informatique
Information sur le livre :
QA203. Calcul a plusieurs variables partie 1. 2011. 3 ans et plus. Auteur : Cay HORTSTMANN; Genre : Informatique
Information sur le livre :
QA204. Calcul a plusieurs variables partie 2. 2011. 3 ans et plus. Auteur : Cay HORTSTMANN; Genre : Informatique
Information sur le livre :
AC409. Le chateau d'Ortrante. 1764. 16 ans et plus. Auteur : Cay HORTSTMANN; Genre : Informatique
Information sur le livre :
BD302. Harry Potter et le prisionier d'Azkaban. 1999. 3 ans et plus. Auteur : JK ROWLING; Genre : Science-Fiction
Information sur le livre :
GA404. Big C++. 2016. 8 ans et plus. Auteur : Cay HORTSTMANN; Genre : Informatique
Information sur le Dvd :
D8403. Rush Hour. 1998. 2 ans et plus. Realisateur : Al; Acteurs : Jackie Chan; Chris Tucker;
Information sur le Dvd :
D7203. Avenger. 2012. 3 ans et plus. Realisateur : Nick Fury; Acteurs : Iron Man; Thor; Hulk; Captain America; Black Window;
_____
AJOUT DES DIFFERENTS ELEMENTS A LA BIBLIOTHEQUE
```

```
_____________________________________________
TESTS D'EMPRUNTS

1630236 a reussi 2 emprunt(s) !
1630236 a rate 5 emprunt(s) !
1782965 a reussi 2 emprunt(s) !
1782965 a rate 5 emprunt(s) !
1865487 a reussi 2 emprunt(s) !
1865487 a rate 5 emprunt(s) !
p878546 a reussi 2 emprunt(s) !
p878546 a rate 5 emprunt(s) !
_____________________________________________
INFO ABONNE AVANT RETOUR

Albus, Dumbledore. 78 ans. #p878546
LISTE DE LIVRES :
1 - Usager #p878546. Livre Information sur le livre :
BD302. Harry Potter et le prisionier d'Azkaban. 1999. 3 ans et plus. Auteur : JK ROWLING; Genre : Science-Fiction
Retour prevu : 160204

2 - Usager #p878546. Dvd Information sur le Dvd :
D7203. Avenger. 2012. 3 ans et plus. Realisateur : Nick Fury; Acteurs : Iron Man; Thor; Hulk; Captain America; Black Window;
Retour prevu : 160204

Abonne - 1876458 - non trouve
_____________________________________________
TESTS RETOUR LIVRE

BD302 remis par 1630236
Echec remise
Echec remise
Echec remise
D7203 remis par p878546
Echec remise
Echec remise
_____________________________________________
INFO ABONNE APRES RETOUR

Albus, Dumbledore. 78 ans. #p878546
LISTE DE LIVRES :
1 - Usager #p878546. Livre Information sur le livre :
BD302. Harry Potter et le prisionier d'Azkaban. 1999. 3 ans et plus. Auteur : JK ROWLING; Genre : Science-Fiction
Retour prevu : 160204
```

```
_____________________________________________
INFO ABONNE APRES RETOUR

Albus, Dumbledore. 78 ans. #p878546
LISTE DE LIVRES :
1 - Usager #p878546. Livre Information sur le livre :
BD302. Harry Potter et le prisionier d'Azkaban. 1999. 3 ans et plus. Auteur : JK ROWLING; Genre : Science-Fiction
Retour prevu : 160204

_____________________________________________
ENTREZ UN MOT A RECHERCHER (1er test)

c
Recherche pour le mot : c
Information sur le livre :
GA404. Big C++. 2016. 8 ans et plus. Auteur : Cay HORTSTMANN; Genre : Informatique
Information sur le livre :
QA203. Calcul a plusieurs variables partie 1. 2011. 3 ans et plus. Auteur : Cay HORTSTMANN; Genre : Informatique
Information sur le livre :
QA204. Calcul a plusieurs variables partie 2. 2011. 3 ans et plus. Auteur : Cay HORTSTMANN; Genre : Informatique
Information sur le livre :
AC409. Le chateau d'Ortrante. 1764. 16 ans et plus. Auteur : Cay HORTSTMANN; Genre : Informatique
Information sur le livre :
BD302. Harry Potter et le prisionier d'Azkaban. 1999. 3 ans et plus. Auteur : JK ROWLING; Genre : Science-Fiction
Information sur le Dvd :
D8403. Rush Hour. 1998. 2 ans et plus. Realisateur : Al; Acteurs : Jackie Chan; Chris Tucker;
Information sur le Dvd :
D7203. Avenger. 2012. 3 ans et plus. Realisateur : Nick Fury; Acteurs : Iron Man; Thor; Hulk; Captain America; Black Window;
_____________________________________________
ENTREZ UN MOT A RECHERCHER (2e test)

al
Recherche pour le mot : al
Information sur le livre :
QA203. Calcul a plusieurs variables partie 1. 2011. 3 ans et plus. Auteur : Cay HORTSTMANN; Genre : Informatique
Information sur le livre :
QA204. Calcul a plusieurs variables partie 2. 2011. 3 ans et plus. Auteur : Cay HORTSTMANN; Genre : Informatique
Information sur le Dvd :
D8403. Rush Hour. 1998. 2 ans et plus. Realisateur : Al; Acteurs : Jackie Chan; Chris Tucker;
_____________________________________________
ENTREZ UN MOT A RECHERCHER (3e test)

bob
Recherche pour le mot : bob
Aucun Resultat Trouve :-(
Appuyez sur une touche pour continuer...
```

Question

1. Quel est la différence entre définir un attribut comme public, protected et private ?
2. Quel(s) problème(s) constatez-vous au niveau des informations qui sont affichées sur l'abonné Albus ? D'où vient ce problème ?
3. Quel est le problème avec la valeur affichée pour la limite d'emprunts de l'étudiant au baccalauréat de Polytechnique Montréal ? Donnez une solution pour résoudre cela.

Correction

La correction du TP2 se fera sur 20 points.

Voici les détails de la correction :

- (03 points) Compilation du programme;
- (03 points) Exécution du programme;
- (04 points) Comportement exact des méthodes du programme;
- (04 points) Utilisation adéquate de l'héritage;
- (02 points) Utilisation correcte de la portée des attributs;
- (01 point) Documentation du code;
- (01 point) Utilisation correcte du mot-clé *this* et const;
- (02 points) Réponses aux questions.