COLLÈGE MONTMORENCY

Travail pratique #3

Pondération 20%

Date de remise : voir Moodle

Ce travail pratique sera validé par un test en classe (quiz) qui aura lieu après la remise du travail. Vous recevrez une seule, note qui comptera pour 20% de la note finale.

Une pénalité de 10% sera appliquée si les directives de remise ci-dessous ne sont pas respectées.

- Créez un package nommé **`tp03` dans le package qui porte votre nom.** Ce package va contenir toutes les classes de ce travail pratique.
- Déposez le package `tp03` compressé en format .zip dans Moodle.
- Chaque classe doit avoir un en-tête selon le modèle ici-bas. N'oubliez pas d'inscrire votre nom après le tag `@author`

```
/**
 * Description de la classe.
 *
 * @author votre nom
 *
 */
```

Barème de correction

	Maximum	Note obtenue
Programmes Java : Les classes • Respect des normes de programmation Java • Respect de la convention de nommage des méthodes et des variables • Code clair et efficace	75	
Exécution • Les résultats sont exacts • La présentation des résultats est comme demandée	25	
Total	100	

Mise en situation

Poste Canada vous demande de développer une application en java pour gérer le courrier.

Poste Canada permet de livrer différents types de courrier : lettre, colis et prospectus.

- **Une lettre** est caractérisée par un *poids* (en g) qui doit être inférieur à 500, une *option d'envoi* (régulier, rapide), *une adresse du destinataire* (chaine de caractères), *et une adresse de l'expéditeur* (chaine de caractères).
- **Un colis** est caractérisé par un *poids* (en g) qui doit être supérieur à 500g et inférieur 50000g, *une option d'envoi* (régulier, rapide), *une adresse du destinataire* (chaine de caractères), *une adresse de l'expéditeur* (chaine de caractères), *une largeur* (en cm), une *hauteur* (en cm), le *tarif de base initialisé à 7,59$ et le tarif unitaire par unité de poids initialisé à 0,35$. Ces deux tarifs permettent de calculer le tarif d'affranchissement comme expliqué ci-dessous.*
- **Un prospectus:** il s'agit de documents de publicité comme les circulaires, les journaux, et les dépliants**.** Un prospectus est caractérisé par *un poids* (en g) qui doit être inférieur à 200g, *une option d'envoi* (régulier, rapide), *une région de distribution* (les 3 premiers caractères du code postal comme J7B) et *un titre* (une chaine de caractères).

Travail demandé

On veut implémenter les classes nécessaires pour gérer les différents types de courrier en utilisant l'héritage. La ***figure 1*** représente la hiérarchie des classes qu'on a ressortie de la mise en situation. En plus, il faut implémenter une classe ***ListeCourrier*** et une classe ***GestionCourrier*** (décrites ci-dessous) pour gérer les courriers.

La classe Courrier (voir ***figure 1***) contiendra :

- Deux constantes : **`ENVOI_REGULIER = 1`** et **`ENVOI_RAPIDE = 2`**
- Les attributs d'instances : poids et optionEnvoi.
- Un constructeur à 2 paramètres.
- Une méthode ***`toString()`*** qui retourne une description du courrier sous forme d'une chaine de caractères .
- Une méthode abstraite ***`calculerTarif()`***.

Il faut implémenter les classes `Lettre`, `Colis` et `Prospectus` en respectant la hiérarchie proposée ici-bas.

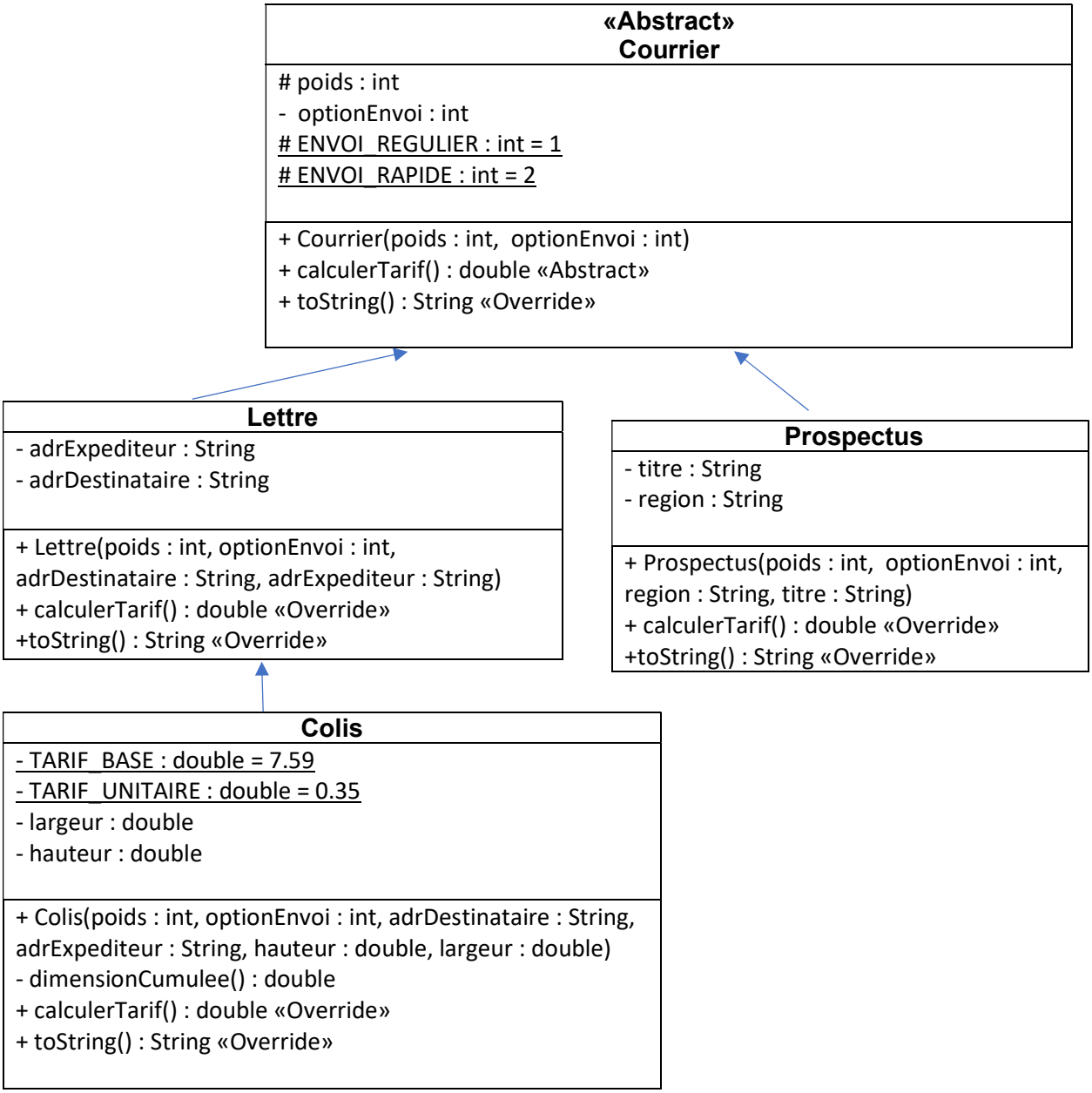

Figure 1 : Hiérarchie des classes

Notation:

- Les attributs et méthodes statiques sont soulignés.
- +public.
- -private
- #protected
- Les classes et les méthodes abstraites sont représentées par le stéréotype «abstract»
- Les méthodes redéfinies sont représentées par le stéréotype «Override»

Il faut respecter les règles suivantes :

- Une classe ne comportera que les méthodes/attributs qui lui sont spécifiques
- Les modificateurs d'accès devront être clairement spécifiés.
- Vos classes doivent éviter de dupliquer inutilement des méthodes ou des attributs, utilisez plutôt l'héritage et ses avantages : la redéfinition, l'appel de la méthode de la classe supérieure, etc.

Il faut implémenter au moins :

- Les constructeurs nécessaires qui valident les données (attributs d'instance).
- Les modificateurs et les accesseurs.
- Une méthode ***`toString()`*** qui retourne la description du courrier sous forme d'une chaine de caractères. **Cette méthode doit être redéfinie dans chacune des sous classes**. Pour une lettre, la chaine sera formée de l'adresse de l'expéditeur, l'adresse du destinataire, le poids, l'option d'envoi et le tarif. Pour un colis, la chaine sera formée de l'adresse de l'expéditeur, l'adresse du destinataire, le poids, l'option d'envoi, le tarif, la largeur, la hauteur et la dimension cumulée. Pour un prospectus, la chaine sera formée par le poids, l'option d'envoi, le tarif, le titre et la région. L'affichage doit se faire comme dans l'exemple suivant :

```
Lettre :
 Adresse de l'expediteur : 38 rue des Oiseaux H5R3B2
 Adresse du destinataire : 41 rue des Pinçons H8B2T6
 Poids : 55
 Option d'envoi : regulier
 Tarif : 1,94$

Colis :
 Adresse de l'expediteur : 13 rue Judith H6R3C4
 Adresse du destinataire : 8 rue Péloquin H2B6G5
 Poids : 750
 Option d'envoi : rapide
 Tarif : 650,18$
 Largeur: 90.0
 Hauteur: 150.0
 Dimension cumulee : 240.0

Prospectus:
 Poids : 50
 Option d'envoi : regulier
 Tarif : 0,35$
 Titre : Gym Fitness
 Region : H2B
```

- Une méthode ***`calculerTarif()`*** pour calculer le tarif d'envoi du courrier (voir la règle de calcul ici-bas). **Cette méthode doit être redéfinie dans chacune des sous classes.**
- La méthode ***`dimensionCumulee()`*** de la classe `Colis` retourne la hauteur + la largeur.

Le **tarif** d'affranchissement est calculé ainsi par Poste Canada :

a. En option d'envoi régulier:

- **Pour une lettre**

Poids	Tarif
Jusqu'à 30 g	1,07 $
Plus de 30 g jusqu'à 50 g	1,30 $
Plus de 50g Jusqu'à 100 g	1,94 $
Plus de 100g Jusqu'à 200g	3,19 $
Plus de 200g jusqu'à 500g	5,47$

- **Pour un colis**

Tarif de base = 7,59$, Tarif unitaire = 0,35$
La dimension cumulée = largeur + hauteur
La formule de calcul :

Si (dimension cumulée <= 200) alors
    Tarif = Tarif de base + tarif unitaire * poids
Sinon
    Tarif = Tarif de base + tarif unitaire * poids + 55$
Fin Si

- **Pour un Prospectus**

Poids	Tarif
Jusqu'à 50 g	0,35 $
Plus de 50 g jusqu'à 200 g	0,65 $

b. **En option d'envoi rapide** : les tarifs précédents sont **doublés** pour tous les types de courrier.

Ajoutez une classe **ListeCourrier.** Ses principaux attributs sont :

- ***tabCourrier*** : un tableau qui contient les courriers reçus dans le bureau de poste.
- ***nbCourriers*** : le nombre de courriers dans le tableau ***tabCourrier*.**

Ses méthodes principales sont :

1. Un constructeur `public ListeCourrier(int maxCourriers)` qui consiste à :
    a. initialiser le tableau `tabCourrier` avec un nombre de cases égal à `maxCourriers`.
    b. Initialiser `nbCourriers` à zéro.

2. Une méthode `public boolean ajouterCourrier (Courrier unCourrier)` qui permet de rajouter un courrier si le tableau ***tabCourrier*** n'est pas plein. Elle retourne ***true*** si le courrier est ajouté et ***false*** sinon.

3. Une méthode `public double calculerSommeTarifCourrier()` qui calcule et retourne la somme des tarifs d'envoi de tous les courriers dans ***tabCourrier***.

4. Une méthode `public void afficherCourrier(int typeCourrier)` qui affiche les courriers dans ***tabCourrier*** selon la valeur du paramètre.

- Si typeCourrier vaut 1, elle affiche toutes les lettres,
- Si typeCourrier vaut 2, elle affiche tous les colis,
- Si typeCourrier vaut 3, elle affiche tous les prospectus,
- Si typeCourrier vaut 4, elle affiche tous les courriers (tous les types).

Note: pour afficher toutes les lettres sans afficher les colis, vous devez vérifier que tabCourrier[i] est une instance de **Lettre** et n'est pas une instance de **Colis** sinon tous les colis sont affichés également puisqu'un colis est une lettre.

5. Une méthode `public boolean estVide()` **:** retourne ***true*** si ***tabCourrier*** est vide.
6. Une méthode `public boolean estPlein()` **:** retourne ***true*** si ***tabCourrier*** est plein.
7. Une méthode `public int taille()` **:** retourne le nombre de courriers dans ***tabCourrier***.

Ajouter une classe **GestionCourrier** qui permettra de gérer le courrier. Cette classe contient une variable de type Scanner qui permet la lecture à partir du clavier.

```
public static Scanner clavier = new Scanner(System.in);
```

Elle contient les méthodes statiques suivantes que vous devez implémenter :

1. `public static void main(String[] args)`**:** Cette méthode déclare et initialise une liste de 100 courriers (`listeCourrier`). Elle appelle la méthode `initialiserListeCourrier`() pour initialiser la liste des courriers (voir la méthode suivante) puis affiche le menu suivant :

   GESTION DES COURRIERS

   1. Afficher les courriers
   2. Ajouter un courrier
   3. Afficher la somme de tous les tarifs des courriers
   4. Quitter

   Entrez votre choix:

   Ce menu doit s'afficher tant que l'utilisateur n'a pas choisi l'option 4 (pour quitter l'application).

   Pour chaque option du menu, vous devez effectuer le traitement approprié en appelant la méthode appropriée.

2. `public static void initialiserListeCourrier(ListeCourrier liste)`**:** Cette méthode ajoute les 7 courriers suivants dans la liste des courriers:

	(Poids,	**Option d'envoi,**	**Adr destinataire,**	**Adr expéditeur)**
Lettre1 :	25,	envoi rapide,	"60 rue des Tulipes H7Y3S5",	"32 rue des Lilas H3B4J5"
Lettre2:	55,	envoi régulier,	"41 rue des Pinçons H8B2T6",	"38 rue des Oiseaux H5R3B2"
Lettre3:	36,	envoi rapide,	"12 rue Dagenais H6D5A4",	"9 rue Delhi G2X3R2"

(Poids, option d'envoi, adr destinataire, Adr expéditeur, hauteur, largeur)

Colis1: 550, envoi régulier, "24 rue Drapeau H1D3F5", "45 rue Renaissance H8Y3S5", 100.0, 50.0

Colis2: 750, envoi rapide, "8 rue PéloquinH2B6G5", "13 rue Judith H6R3C4", 150.0, 90.0

	(Poids,	**option d'envoi,**	**titre**	**region)**
Prospectus1 :	30,	envoi rapide,	"Resto FuPam",	"H7X"
Prospectus2 :	50,	envoi régulier,	"Gym Fitness",	"H2B"

3. `public static void afficherCourrier(ListeCourrier liste)`:
Cette méthode doit afficher un menu pour choisir le type de courrier à afficher :

AFFICHER LES COURRIERS
Entrez:
1. **pour afficher toutes les lettres,**
2. **pour afficher tous les colis,**
3. **pour afficher tous les prospectus**
4. **pour afficher tous les courriers:**

Quel est votre choix?

Si le choix entré est erroné, le message d'erreur « `Choix invalide` » doit être affiché.

4. `public static void ajouterCourrier(ListeCourrier liste)` :
Cette méthode doit afficher un menu pour choisir le type de courrier à rajouter :

AJOUT D'UN COURRIER
Entrez:
1. **pour Lettre,**
2. **pour Colis,**
3. **pour Prospectus:**

Quel est votre choix?

Si le choix entré est erroné, le message d'erreur « `Choix invalide` » doit être affiché.
Si le choix est valide (1, 2 ou 3), la méthode demandera ensuite à l'utilisateur de rentrer les informations du nouveau courrier à partir du clavier. Elle crée un objet `Courrier` avec ces informations qu'elle ajoute ensuite dans la liste quand c'est possible.

Si la liste est pleine, le message d'erreur suivant sera affiché : «`Impossible d'ajouter le courrier, la liste est pleine`».

Si le courrier est ajouté, un message de confirmation sera affiché sous forme: «`Le courrier a ete ajoute avec succes`».

NB : Rajouter les méthodes que vous voyez nécessaires. Utiliser les modificateurs adéquats à chaque situation (***static***, ***final***, ***private***, ***public***, …).