TP 2 : Programmation orientée objet

Exercice 1 – Classe Domino

Définissez une classe Domino qui permet d'instancier des objets simulant les pièces d'un jeu de dominos. Le constructeur de cette classe initialisera les valeurs des points présents sur les deux faces A et B du domino (valeurs par défaut = 0).

Définissez les attributs, les constructeurs, les accesseurs/mutateurs ainsi que les méthodes associées à la classe.

Deux autres méthodes seront définies :

• Une méthode **affichePoints()** qui affiche les points présents sur les deux faces ;

• Une méthode **sommePoints()** qui renvoie la somme des points présents sur les 2 faces.

Exercice 2 – Classe compte bancaire

Définissez une classe **CompteBancaire**, qui permet d'instancier des objets tels que compte1, compte2, etc. Le constructeur de cette classe initialisera trois attributs d'instance nom, RIB et solde. Par défaut l'ouverture à l'ouverture du compte le client doit disposer de 100 DHS sur son solde.

Définissez les attributs, les constructeurs, les accesseurs/mutateurs ainsi que les méthodes associées à la classe.

Trois autres méthodes seront définies :

• **depot(**somme**)** permettra d'ajouter une certaine somme au solde ;

• **retrait(**somme**)** permettra de retirer une certaine somme du solde ;

• **afficheSolde()** permettra d'afficher le nom du titulaire et le solde de son compte.

Exercice 3 : Classe Satellite

Définissez une classe **Satellite** qui permette d'instancier des objets simulant des satellites artificiels lancés dans l'espace, autour de la terre. Le constructeur de cette classe initialisera les attributs d'instance suivants, avec les valeurs par défaut indiquées : **nom**, **masse = 100, vitesse = 0.**
Lorsque l'on instanciera un nouvel objet Satellite, on pourra choisir son nom, sa masse et sa vitesse.
Les méthodes suivantes seront définies :

- **impulsion(force, duree)** permettra de faire varier la vitesse du satellite. Pour savoir comment, rappelez-vous votre cours de physique : la variation de vitesse $\Delta \mathbf{v}$ subie par un objet de masse **m** soumis à l'action d'une force **F** pendant un temps **t** vaut :

$\Delta v = \frac{F X t}{m}$. Par exemple : un satellite de 300 kg qui subit une force de 600 Newtons pendant 10 secondes voit sa vitesse augmenter (ou diminuer) de 20 m/s.

• **afficheVitesse()** affichera le nom du satellite et sa vitesse courante.

• **energie()** renverra au programme la valeur de l'énergie cinétique du satellite.

Rappel : l'énergie cinétique se calcule à l'aide de la formule : $Ec = \frac{m*v*v}{2}$

Exercice 4 – Voiture et Conducteur

Définissez une classe Voiture qui permet d'instancier des objets reproduisant le comportement de voitures automobiles. Le constructeur de cette classe initialisera les attributs d'instance suivants, avec les valeurs par défaut indiquées :

marque = 'Ford', couleur = 'rouge', pilote = Conducteur, vitesse = 0.

Lorsque l'on instanciera un nouvel objet Voiture, on pourra choisir sa marque et sa couleur, mais pas sa vitesse, ni le nom de son conducteur.

Définissez les attributs, les constructeurs, les accesseurs/mutateurs ainsi que les méthodes associées aux classes Conducteur et Voiture.

Les méthodes suivantes seront définies :

• **accelerer(taux, duree)** permettra de faire varier la vitesse de la voiture. La variation de vitesse obtenue sera égale au produit : taux × duree. Par exemple, si la voiture accélère au taux de 1,3 m/s pendant 20 secondes, son gain de vitesse doit être égal à 26 m/s. Des taux négatifs seront acceptés (ce qui permettra de décélérer). La variation de vitesse ne sera pas autorisée s'il n'ya pas de conducteur dans la voiture.

• **afficheProprietе()** permettra de faire apparaître les propriétés présentent de la voiture, c'est-à-dire sa marque, sa couleur, le nom de son conducteur, sa vitesse.

Exercice 5 – Calcul de la surface d'un terrain

Une compagnie immobilière souhaite mener des opérations de délimitation des parcelles de terrain pour la construction des logements sociaux sur la ville verte de Bouskoura Golf City. Pour cela, elle aimerait mettre en place un programme permettant le calcul la surface totale des terrains exploités ainsi que le coût de vente d'un terrain.

Le calcul de la surface du terrain est défini comme suit :

- La surface d'un terrain est composée de deux rectangles.
- Un rectangle est caractérisé par sa longueur et largeur.
- La surface *S* du rectangle est calculée comme suit : *S = longueur x largeur*

- Ainsi la surface totale du terrain est la somme de la surface des deux rectangles composés.
- Le prix de vente d'une parcelle de terrain est de 3000dhs/m^2

Exercice 6 : Gestion de pharmacie

Écrivez un programme orienté objets qui permet de gérer une pharmacie.
La pharmacie gère des clients et des médicaments. Un client est caractérisé par un nom et un crédit. Le crédit représente la somme que ce client doit à la pharmacie. Le crédit peut être négatif si le client a versé plus d'argent que le montant. Un médicament est caractérisé par un nom (de type String), un prix (de type double) et un stock (de type int). On souhaite àvoir une applicaion qui propose les option suivantes :

- **affichage(..)** permet d'afficher les clients et leurs crédits respectifs ainsi que les médicaments et leurs stocks respectifs.

- **approvisionnement(..)** permet d'approvisionner le stock d'un médicament. Le nom du médicament à approvisionner ainsi que la quantité à ajouter au stock doivent être lus depuis le terminal. Lorsque le nom du médicament est introduit, il faut vérifier qu'il s'agit bien d'un nom connu dans la liste des médicaments de la pharmacie. Si le médicament se trouve dans la liste alors on incrémente son stock.
- **enregistrement(...)** permet d'ajouter un nouveau médicament dans la liste des médicaments de la pharmacie

- **achat(..)** permet de traiter un achat fait par un client. L'achat porte sur un médicament donné dans une quantité donnée. Pour cette transaction le client paie un certain prix. Une opération d'achat aura pour effet de déduire la quantité achetée du stock du médicaments correspondant et d'augmenter le crédit du client (d'un montant équivalent au montant de l'achat moins la somme payée).
  Les noms du client et du médicament doivent être lus depuis le terminal. Le programme doit boucler jusqu'à introduction de noms connus aussi bien pour les clients que les médicaments. Ces procédures de vérification seront prises en charge par les méthodes lireClient et lireMedicament (voir plus bas). La quantité achetée et le montant payé sont aussi lus depuis le terminal. Ils seront supposés corrects.

- **quitter(..)** affiche le message ``programme terminé!''.

Vous définirez une méthode auxiliaire lireClient(..) prenant comme paramètre un tableau de clients. Elle permettra de lire le nom d'un client depuis le terminal et de vérifier si ce client existe dans le tableau des clients. Dans ce cas le client sera retourné. Cette méthode doit boucler jusqu'à ce qu'un client soit trouvé. Elle sera utilisée par la méthode achat(..). Une méthode similaire, lireMedicament(..) sera fournie pour les médicaments. Elle sera utilisée par les méthodes achat(..) et approvisionnement(..).

Vous êtes libre de définir, en plus de ces méthodes, toutes celles que vous jugerez nécessaires.

Lors de la programmation, il vous est demandé de respecter scrupuleusement les indications suivantes:
Votre programme doit être bien modularisé à la fois sous forme de méthodes auxiliaires de la méthode main() et sous forme de classes pour les objets du programme. En particulier :
Chaque variable et méthode doit être déclarée dans la classe la plus adaptée.

Dans chaque classe liée à un objet, il faut utiliser le mot-clé private autant que possible. En particulier, toutes les variables d'instances seront privées.