TP POO avec Python

2ème année licence informatique et vidéoludisme

Université Paris 8

Exercice 1: *Gestion de compte bancaire*

Un compte bancaire est défini par son numéro, le nom du propriétaire et le solde disponible.

1. Créer une classe *CompteBancaire* ayant pour attributs : *numeroCompte* (type numérique) , *nom* (type chaîne de caractères), *solde* (type réel).
2. Créer un constructeur qui permet de créer un compte bancaire et de l'initialiser.
3. Créer une méthode *crediter()* qui crédite le compte du montant *x* (réel positif).
4. Créer une méthode *debiter()* qui débite le compte du montant *x* (réel positif).
5. Créer une méthode *virementVers()* qui effectue un virement vers un autre compte bancaire.
6. Créer une méthode *afficher()* permettant d'afficher les détails d'un compte bancaire.
7. Tester votre code.

Exercice 2: *Gestion des résultats d'examen*

On désire réaliser un programme qui gère les résultats d'examen d'un ensemble d'étudiants. Chaque étudiant sera représenté par une classe *Etudiant*, comportant les attributs suivants:

- ➔ Nom (type chaîne de caractères)
- ➔ Liste de notes (contenant 3 notes d'examen)
- ➔ Admissibilité à l'examen (un booléen ayant les valeurs suivantes: *False* (non admis) et *True* (admis))

1. Définir la classe *Etudiant* avec ses attributs et son constructeur.

   Pour chaque étudiant, le programme doit lire au clavier le nom de l'étudiant et 3 notes, puis le constructeur en calcule la moyenne et renseigne convenablement le champ d'admissibilité suivant la règle suivante :

   - ➔ moyenne < 10 : non admis
   - ➔ moyenne >= 10 : admis

Nous supposerons que les étudiants sont fournis par une liste placée comme attribut de classe.

2. Créer une méthode de classe *ajouterEtudiant()* qui permet d'ajouter un nouvel étudiant à la liste. Modifier le constructeur de la classe pour que l'instance créée soit ajoutée à la liste.
3. Créer une méthode *afficher()* qui affiche le nom de l'étudiant et son admissibilité.
4. Créer une méthode de classe *afficherResultat()* qui affiche les résultats de tous les étudiants.
5. Dans le programme principal, tester votre code en créant 3 étudiants et en affichant les résultats. Voici un exemple d'exécution:

    Resultats de l'examen:

    Edward Elric - Admis

    Anya forger - Non admis

    Jean Kirstein - Non admis

Exercice 3: *Gestion des salles*

Une entreprise souhaite informatiser la gestion de ses salles de réunion, et mettre en place un système de réservation par internet.

1. Définir une classe *Salle* avec les attributs suivants : *id*, *nom*, et *disponibilite*.

    La disponibilité d'une salle est représentée par une matrice 5x6 dont les lignes représentent les jours de la semaine (du lundi au vendredi) et les colonnes représentent les créneaux (9h-10h, 10h-11h, 11h-12h, 15h-16h, 16h-17h, 17h-18h). Si la salle est réservée pendant un créneau, la valeur de la case est *True*.

2. Définir un constructeur permettant d'initialiser les attributs de la classe *Salle*, sachant que *id* doit être auto-incrément.
3. Créer une méthode *reserver()* qui permet à un employé de réserver une salle en choisissant une date et un créneau.
4. Créer une méthode *annulerRes()* qui permet à un employé d'annuler la réservation d'une salle.
5. Définir la méthode *afficherSalle( )* permettant d'afficher les informations d'une salle.
6. Définir une classe *SalleMachine* qui hérite de la classe Salle. Cette classe doit deux champs supplémentaires:

- ➔ *nombreMachines* (type numérique): qui désignent le nombre de machines disponibles dans cette salle.
- ➔ *videoProjecteur* (type booléen): qui est vrai si la salle dispose d'un vidéo projecteur.

7. Surcharger la méthode *afficherSalle()* pour afficher le matériel disponible dans une salle machine.
8. Pour tester votre code :
    a. Créer une salle normale.
    b. Afficher les informations de la salle.
    c. Réserver la salle.
    d. Annuler la réservation.
    e. Créer une salle machine.
    f. Afficher les informations de la salle.