TP PROGRAMMATION ORINTEE OBJET PYTHON

Exercice 1.

1. Les classes sont souvent utilisées pour modéliser des objets dans le monde réel. Nous pouvons représenter les données sur une personne dans un programme par une classe Personne, contenant le nom de la personne, son prénom, son numéro de téléphone, et son email. Une méthode __str__ peut imprimer les données de la personne.

   Exemples :

   >>> amir = Personne('Ben Salem', 'Amir', '55593581', "amirbs@ipein.rnu.com")
   >>> print(amir)
   Ben Salem, Amir --Telephone: 55593581 --Email: "amirbs@ipein.rnu.com"
   >>> mariem = Personne('Abassi','Mariem','55519403',"mariem12@gmail.com")
   >>> print(mariem)
   Abassi, Mariem --Telephone : 55519403 --Email :mariem12@gmail.com

   Travail à faire: Implémentez la classe Personne

2. Un travailleur est une personne ayant un emploi. Dans un programme, un travailleur est naturellement représenté comme une classe Travailleur dérivée de la classe Personne, parce qu'un travailleur est une personne, c'est-à-dire, nous avons une relation est-un. La classe Travailleur étend la classe Personne avec des données supplémentaires, par exemple le nom de l'entreprise, l'adresse de l'entreprise et le numéro de téléphone du travail. La fonctionnalité 'impression (la méthode spéciale __str__) doit être modifiée en conséquence.
   Travail à faire: Mettre en œuvre cette classe Travailleur.
   Un scientifique est un type spécial de travailleur. La classe Scientifique peut donc être dérivée de la classe Travailleur. Ajouter des données sur la discipline scientifique (physique, chimie, mathématiques, informatique, ...). On peut aussi ajouter le type de scientifique: théorique, expérimental ou informatique. La valeur d'un tel attribut de type ne doit pas être limitée à une seule catégorie, car un scientifique peut être classé comme, par exemple, à la fois expérimental et informatique (c'est-à-dire, vous pouvez représenter la valeur sous la forme d'une liste ou d'un tuple).
   Travail à faire: Mettre en œuvre la classe Scientifique.
3. Enfin, faites un programme principal de démonstration où vous créez et imprimez des instances de classes Personne, Travailleur et Scientifique. Imprimez le contenu des attributs de chaque instance.

Exercice 2:

Dans cet exercice on s'intéresse à créer des classes pour gérer les vols d'une compagnie aérienne locale qui organise des vols entre des villes. Plus précisément on s'intéressera aux plans de vol entre les différentes villes. C.-à-d. les vols disponibles ainsi que l'heure de départ.

1. Pour créer une classe Vol_direct qui représentera un vol direct entre deux villes (pas d'escale dans une ville intermédiaire), on doit:
   1.1. Définir le constructeur de cette classe qui a quatre attributs:
   - Dep et arr qui désigne respectivement la ville de départ et la ville d'arrivée
   - jour qui désigne le jour de la semaine (lundi, mardi, ...)
   - heure(un entier entre 0 et 24 qui représente l'heure de départ)

   1.2. Ecrire une méthode Affiche qui affiche une chaine bien formatée de la forme: « Ce vol part de 'Tunis' vers 'Djerba' le 'lundi' à 9 heure »

2. Créer une classe Vols qui représentera tous les vols le long de la semaine en utilisant la classe Vol_direct. Pour ce faire on doit:
   2.1. Définir le constructeur de cette classe avec un seul attribut qui est une liste de vols
   2.2. Ecrire une méthode Liste_successeurs qui retourne une liste contenant les villes arrivées d'une ville de départ passée comme paramètre
   2.3. Ecrire une méthode Appartient qui vérifie si une ville appartient au plan du vol que ce soit comme ville d'arrivée ou de départ
   2.4. Ecrire une méthode Affiche qui affiche tous les vols directs
3. Ecrire un programme principal permettant de:
   Créer une liste LV d'objets Vol_direct
   a) NB:on suppose avoir définie les 3 fonctions suivantes: Saisie_Jour qui retourne un jour valide, Saisie_Heure() qui retourne une heure valide et Saisie _Ville() qui retourne un nom de ville valide.
   b) Créer un objet Vol nommé V à partir de la liste déjà créée
   c) Afficher tous les vols
   d) Saisir une ville qui doit appartenir au plan du vol puis calculer et afficher la liste de ses successeurs