ITI 1520 - Devoir 6

Disponible: le 23 novembre, 2020
Date de remise: lundi, le 7 décembre, 8:00
SVP notez que le devoir n'est pas accepter après cette date.

Vous devez faire ce travail **individuellement**. Vous devez soumettre un fichier d6_VOTRE_NUMERO_ETUDIANT.zip contant un fichier pour chaque question: d6q1_xxxxxx.py (qui est donné mais vous devrez remplir quelques fonctions dans ce fichier après remplacer xxxxxx dans le nom de ficher avec votre numéro d'étudiant) et d6q2_VOTRE_NUMERO_ETUDIANT.py (avec les trois fonctions requises).

Les noms des fichiers et les noms des fonctions doivent être les noms requises, parce qu'on va utiliser des tests automatiques pour la correction. Si le nom d'une fonction n'est pas le nom requis, la note sera zéro pour la fonction.
Ajoutez des commentaires pour chaque fonction/méthode (avec une description courte, contrat de type et preconditions) s'ils ne sont pas déjà donnés.
Si un fichier .py donne erreur de syntaxe, la note sera zéro pour la question.
Si vous envoyez le devoir plusieurs fois, la dernière version sera note.
SVP noter qu'on va utiliser un outil logiciel pour détecter du plagiat. En cas deux devoirs sont identique ou très similaires, la note sera zéro pour les deux.

Barème : total de 20 points. Devoir 6 est 7% de la note finale.

Question 1 (12 points) Gestion de campus

Dans ce travail, vous devez implémenter les classes: *Personne*, *Etudiant*, *Employe* et *Gestion.*
On fournit le diagramme UML contenant la description de toutes les classes.
Le programme principal est déjà implémenté dans le fichier *d5q1_xxxxxx.py*. Vous devez donc compléter toutes les classes.

Classe Personne
La classe *Personne* comprendra les attributs suivants :

- *nom* est une chaine de caractères.
- *prénom* est une chaine de caractères.
- *identifiant* est un nombre entier.

Cette classe dispose les méthodes suivantes :

- La méthode `__init__()` doit initialiser le nom d'un objet de type classe Personne, son prénom et son identifiant. Cette fonction prend en entrée un nom, un prénom de type chaine de caractères et un identifiant de type entier.
- La méthode `_repr_()` doit retourner une chaine de caractères qui contient tous les informations d'une personne.
- La méthode `_eq_()` doit vérifier si deux personnes sont identiques (le même nom, le même prénom et le même identifiant). Cette fonction elle prend en entrée un objet de type *Personne* et retourne *True* si les deux personnes sont identiques et *False* sinon.

Classe Employe
La classe *Employe* dérive de la classe *Personne*. Cette classe comprendra les attributs suivants :

- *tauxHoraire* est un nombre réel.

Cette classe dispose les méthodes suivantes :

- La méthode `__init__()` doit initialiser un objet de type *Employe*. Cette fonction prend en entrée un nom, un prénom de type chaine de caractères, un identifiant de type entier et un taux de type réel.
- La méthode `_repr_()` doit retourner une chaine de caractères qui contient tous les informations d'un employé.
- La méthode `calculerSalaire()` doit retourner le salaire d'un employé. Elle prend en paramètre le nombre des heures (un nombre entier). Le salaire égale le nombre des heures multiplié par le taux horaire.

Classe Etudiant

La classe *Etudiant* dérive de la classe *Personne*. Cette classe comprendra les attributs suivants :

- *solde* est un nombre réel (présente le solde à payer pour les frais de scolarité).
- *cours* est une liste de cours (une liste de chaine de caractères).

Cette classe dispose les méthodes suivantes :

- La méthode `__init__()` doit initialiser un objet de type *Etudiant*. Cette fonction prend en entrée un nom, un prénom de type chaine de caractères, un identifiant de type entier, un solde de type réel et une liste de cours. La liste de cours doit être initialisée à une liste de chaine de caractères vide.
- La méthode `_repr_()` doit retourner une chaine de caractères qui contient toutes les informations d'un étudiant. Pour la liste de cours, ajouter seulement le nombre de cours dans la chaine de caractères.
- La méthode `ajouterCours()` doit ajouter un cours dans la liste de cours. Elle prend en paramètre le nom de cours (une chaine de caractères). Un étudiant peut ajouter un cours seulement si il n'a aucun solde à payer (la valeur de solde est égale à zéro). Cette fonction retourne *True* si le cours a été ajouté dans la liste de cours et *False* sinon.

Classe Gestion

La classe *Gestion* permet de gérer la liste des étudiants et des employés. Cette classe comprendra deux attributs (statiques) :

- Une liste des étudiants *listEtudiant*.
- Une liste des employés *listEmploye*.

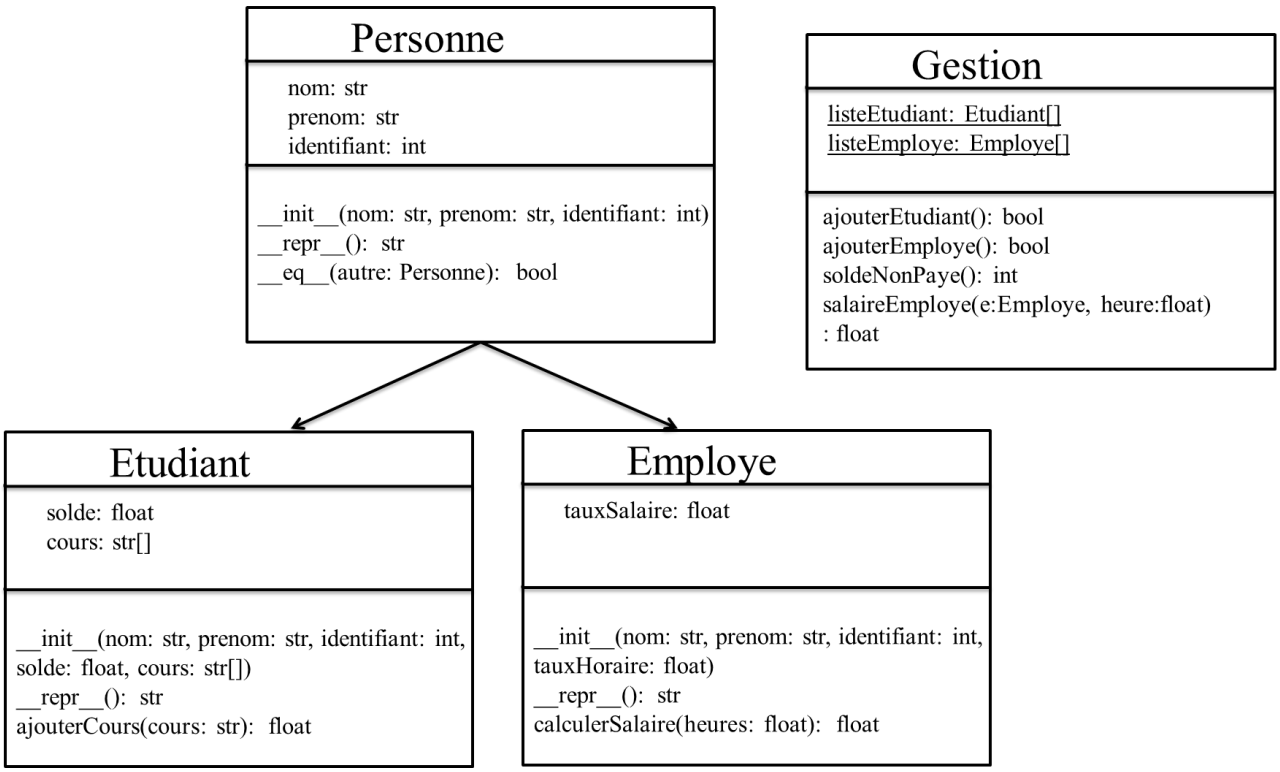
Cette classe ne comprend pas aucuns attributs.

Cette classe dispose les méthodes suivantes :

- La méthode `ajouterEtudiant()`. Cette fonction ne prend aucun paramètre en entrée. Elle permet de demander à l'utilisateur les informations pour l'étudiant à ajouter. Par la suite, elle doit demander à l'utilisateur d'entrer un choix : si l'utilisateur veut ajouter un cours (le choix de l'utilisateur doit être oui ou non, sinon afficher un message approprié). Avant d'ajouter l'étudiant, il faut vérifier qu'il n'existe pas dans la liste. Cette fonction elle doit retourner *True* si l'étudiant a été ajouté et *False* sinon.
- La méthode `ajouterEmploye()`. Cette fonction ne prend aucun paramètre en entrée. Elle permet de demander à l'utilisateur les informations pour l'employé à ajouter. Avant d'ajouter l'employé, il faut vérifier qu'il n'existe pas dans la liste. Cette fonction elle doit retourner *True* si l'employé a été ajouté et *False* sinon.
- La méthode `soldeNonPaye()`. Cette fonction ne prend aucun paramètre en entrée. Elle permet de calculer et retourner le nombre des étudiants qui ont un solde non payé.

- La méthode `salaireEmploye()`. Cette fonction prend en entrée un employé et le nombre des heures travaillées. Elle permet de calculer et retourner le salaire d'un employé. Si l'employé n'existe pas dans la liste, le salaire est égal à zéro.

Le diagramme UML suivant représente la description des classes.



Question 2 (8 points)

a. Écrivez une fonction Python **récursive** appelée `triangle` qui prendra comme paramètre un entier non négatif et qui générera un dessin composé d'étoiles tel qu'affiché ci-dessous. Vous pouvez utiliser une boucle pour générer une ligne d'étoiles, mais pas le dessin en entier.

Exemple :

```
>>> triangle(5)
*
**
***
****
*****
```

b. Écrivez une fonction Python **récursive** appelée `prodListePos_rec` qui prendra comme paramètre une liste et comme deuxième paramètre le nombre des éléments de la liste, et qui retournera le produit des éléments positifs (> 0).

Exemple :

```
>>> l = [1,-2,5,0,6,-5]
>>> prodListePos_rec(l, len(l))
30
```

C. Écrivez une fonction Python récursive appelée `palindrome` qui prendra comme paramètre une chaine des caractères et, et qui retournera True si la chaine est une palindrome ou False sinon. La chaine est un palindrome si le premier caractère est égal avec le dernier caractère, le deuxième caractère est égal avec le caractère avant le dernier, etc. Vous pouvez utiliser des fragments des chaines pour cette question.

Exemples :

```
>>> palindrome("abcba")
True
>>> palindrome("anna")
True
>>> palindrome([])
True
>>> palindrome("abcdabcd")
False
```

Note: Si les fonctions pour Q2 a, b, c **ne sont pas récursives**, le nombre de points reçus sera zéro pour la fonction. (Vous pouvez tester des fonctions équivalentes itératives pour vous-même.)
Testez avec programme principale ou dans l'interpréteur pour vous même. Pas besoin d'envoyez des tests.

Notes pour le fichier declaration_VOTRE_NOM.txt si vous avez besoin:
Ce fichier doit contenir la référence pour code qui n'est pas écrit par vous même **si c'est le cas**. Ça inclue code donné par un collègue ou autre personne, ou trouvé sur l'internet, réseaux sociale (comme Stack Overflow, chegg, discord, ou autre). Ça n'inclue pas le code donne en BrightSpace dans les notes de classe, labo, etc.
Pour chaque question ou vous avez utilisé du code donné ou trouvé, il faut:
1. le numéro de question
2. copier-coller le code emprunter. Ça inclue code donné/trouvé et modifier très peu.
3. le nom de la source: personne, site Internet ou autre
Vous allez perdre des points la question, mais au moins vous évitez une accusation de plagiat. En cas de plagiat, la note est zéro et le cas doit être rapporté au doyen. Le même est valable si quelqu'un montre son code a un collègue.
Si vous n'utilisez pas cette déclaration, c'est équivalent à déclarer que tous le code a été écrit par vous-même.