Programmation objet : exercices

Exercice 1 :

Implémenter une classe `Compte_bancaire` telle que les instances de `Compte_bancaire` possèdent un seul attribut `solde` pouvant être positif ou négatif et telles qu'une instance `A` dispose de méthodes lui permettant de :

- transférer un montant `X` positif vers une instance `B`

- prélever un montant `X` positif depuis une instance `B`

- renvoyer `"Positif"`, `"Négatif"` ou `"Nul"` lorsqu'on lui demande l'état du compte

Exercice 2 :

Définir une classe `Intervalle` représentant des intervalles de nombres. Cette classe possède deux attributs `a` et `b` représentant respectivement l'extrémité inférieure et l'extrémité supérieure de l'intervalle. Les deux extrémités sont considérées comme incluses dans l'intervalle. Tout intervalle avec `b < a` représente l'intervalle vide.

- Ecrire le constructeur de la classe `Intervalle` et une méthode `est_vide` renvoyant `True` si l'objet représente l'intervalle vide et `False` sinon.

- Ajouter une méthode `__len__` renvoyant la longueur de l'intervalle (l'intervalle vide a une longueur 0).

- Ajouter une méthode `__contains__` permettant de tester l'appartenance d'un élément `x` à un intervalle `ivl` avec l'instruction `x in ivl`.

- Ajouter une méthode `intersection` permettant de renvoyer l'intersection de deux intervalles `ivl_1` et `ivl_2` avec l'instruction `ivl_1.intersection(ivl_2)` ou, indifféremment, avec l'instruction `ivl_2.intersection(ivl_1)`.

- (***) Même question avec l'`union` de deux intervalles.

Exercice 3 :

Définir une classe `Duree` pour représenter une durée avec trois attributs : `heures`, `minutes` et `secondes` (où `minutes` et `secondes` sont inférieurs à 60).

- Ecrire son constructeur.

- Ajouter une méthode `__str__` qui renvoie une chaîne de caractères de la forme `"8 h 54 min 36 s"`. Tester en construisant des objets de la classe `Duree` puis en les affichant avec `print`.

- Ajouter une méthode `en_secondes` qui renvoie le nombre de secondes contenues dans une durée.

- Ajouter une méthode `__lt__` qui permet de déterminer si une durée `d_1` est inférieure à une durée `d_2` avec l'instruction `d_1 < d_2`. La tester.

- Ajouter une méthode `tic` qui permet d'augmenter une durée d'une seconde (les attributs minutes et secondes doivent toujours rester inférieurs à 60, on pourra – ou pas – utiliser les opérateurs reste `%` et quotient `//`).

- Ajouter une méthode `multitic` qui permet d'augmenter une durée de `x` secondes (les attributs `minutes` et `secondes` doivent toujours rester inférieurs à 60).