TP 1 - Listes chainées

Ce TP vise à construire la classe **ListeChainee.py** qui correspond à une liste chainée d'entier.

Vous allez devoir créer les différentes méthodes qui constitue une liste chainée.

1 - Premières méthodes des listes chainées

La construction d'une liste se fait de proche en proche à partir de la liste vide.

Exemple : liste1 = ListeChainee(1, ListeChainee(0, None))

TRAVAIL À FAIRE

Cas de la liste vide

- La liste chainée vide correspond à : **ListeChainee(0, None)**
- Implémentez la méthode *estVide* dans le fichier **ListeChainee.py**.
- Exécuter le fichier **TestListeVide.py** qui contient le test unitaire de cette méthode.
  Le test doit être validé.

Cas général

- Définissez maintenant dans le fichier **ListeChainee.py**
- Implémentez le constructeur afin d'initialiser correctement les attributs précédents, puis les accesseurs *getTete* et *getReste*.
- Exécutez le fichier **TestConstruction.py** qui contient les tests unitaires de ces opérations.
  L'ensemble des tests doit être validé.

Définition singleton en Mathématiques : *Ensemble constitué d'un seul élément*

2 Méthodes récursives

Dans cette section, vous allez implémenter les différentes méthodes récursives de la classe **ListeChainee**.
Ces implémentations seront testées par les classes de test fournies.

Vous pouvez également faire vos propres essais en dehors de ces classes de test, en créant un fichier **ProgListeChainee.py** et en y définissant, à votre convenance, des tests manuels (avec des print, par exemple).

TRAVAIL À FAIRE

Parcours récursif

- Implémentez les méthodes *getTaille* et *contient* dans le fichier **ListeChainee.py**.
- Compilez le fichier **TestParcours.java** et exécutez le test. L'ensemble des tests doit être validé.

Conversion en chaîne de caractères

- Implémentez la méthode *__str__* dans le fichier **ListeChainee.py**.
- Exécutez le fichier **TestConversionChaine.py** pour lancez le test. L'ensemble des tests doit être validé.

Ajout d'un élément après un autre

- Implémentez la méthode *ajouteApres* dans le fichier **ListeChainee.py**.
- Exécutez le fichier **TestAjoutApres.py** pour lancez le test. L'ensemble des tests doit être validé.

Retrait d'un élément

- Codez la méthode *retire* dans le fichier **ListeChainee.py**.
- Exécutez le fichier **TestRetrait.py** pour lancez le test. L'ensemble des tests doit être validé.

Ajout d'un élément avant un autre

- Codez la méthode *ajouteAvant* dans le fichier **ListeChainee.py**.
- Exécutez le fichier **TestAjoutAvant.py** pour lancez le test. L'ensemble des tests doit être validé.

A cette étape, tous les fichiers « test » doivent être réussis.
Cependant, il se peut qu'une modification du code de la classe **ListeChainee** afin de réussir un test fasse que l'un des tests précédents ne réussisse plus.

En conséquence, faites une vérification ultime en lançant à nouveau tous les tests.