TP Partie 1 : Liste chainée

i Dans ce TD, nous allons implémenter les listes chaînées sous forme d'un objet `Maillon`.
On considère entre autres que le maillon de départ représente la tête de liste. Le dernier élément aura pour suivant `None`

Question 1 : Une classe

Créer une classe `Maillon` qui possèdera 2 attributs :

- `valeur` : les données du maillon ;
- `suivant` : soit un autre objet maillon (le maillon suivant) soit None

Exemples :

```
>>> m = Maillon( 3 , Maillon(9, Maillon("Bonjour", None )))
>>> print(m.valeur)
3
>>> print(m.suivant.suivant.valeur)
Bonjour
```

Ecrire le code de la classe.

.....

.....

.....

.....

.....

.....

Question 2 : Une assertion

Pour s'assurer que les maillons soient bien constitués, il faut ajouter une assertion qui permet de tester que l'argument suivant soit bien de type `Maillon` (on pourra utiliser `isinstance`) ou None. L'instruction `assert` au début du constructeur qui teste le type du paramètre suivant et qui renvoie le message ci-dessous s'il n'est pas de type Maillon ou égal à `None`.
Message : `L'argument suivant doit être de type Maillon ou None`

```
>>> M = Maillon(3,9)
AssertionError: L'argument suivant doit être de type Maillon ou None
```

Ecrire le code de l'assertion à ajouter au constructeur de classe.

. . . . . .

. . . . . .

Question 3 : L'affichage

On souhaite pouvoir afficher notre liste chaînée à l'aide d'une simple instruction print. Pour cela, il faut que l'objet Maillon possède la méthode `__repr__` (ou `__str__`). Implémenter la méthode **récursive** `__repr__`, qui permettra d'afficher l'instance de la classe `Maillon` à l'aide d'une instruction `print`

Chaque valeur de la chaîne sera séparée par les caractères `->` (attention : il y a un espace avant et après la flèche).

```
>>> m = Maillon( 3 , Maillon(9, Maillon("Bonjour", None )))
>>> print(m)
3 -> 9 -> Bonjour
```

Ecrire le code de la méthode.

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

Question 4 : La taille

Pour connaître le nombre de chaînon de notre liste, il faut implémenter une méthode `taille`.

Implémenter la méthode **récursive** `taille`, qui ne prend pas de paramètre et qui renvoie le nombre de Maillon de la liste.

Exemple :

```
>>> m = Maillon( 1 , Maillon(2, Maillon(3, None )))
>>> print(m.taille())
3
```

Ecrire le code de la méthode.

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

Question 5 : Le premier maillon

On souhaite maintenant accéder aux valeurs des maillons de la chaîne. Pour commencer, on cherche à afficher la valeur du premier maillon. Implémenter la méthode `premier_maillon`, qui ne prendra pas de paramètre et qui renvoie la valeur du premier maillon de la chaîne.

Exemple :

```
>>> chaine = Maillon(21 , Maillon ( 15, Maillon( 45)))
>>> print(chaine.premier_maillon())
21
```

Ecrire le code de la méthode.

.....

.....

.....

.....

Question 6 : Le dernier maillon

On cherche maintenant à obtenir la valeur du dernier élément de la chaîne. Implémenter la méthode `dernier_maillon`, qui ne prendra pas de paramètre et qui renvoie la valeur du dernier maillon de la chaîne.

Exemple :

```
>>> chaine = Maillon(21 , Maillon ( 15, Maillon( 45)))
>>> print(chaine.dernier_maillon())
45
```

Ecrire le code de la méthode.

.....

.....

.....

.....

.....

.....

Question 7 : Accès à un maillon

Implémenter la méthode `maillon`, qui prendra en paramètre un entier `i` et qui renvoie la valeur du maillon d'indice `i`. Assertion : l'indice doit être compris entre 0 et la longueur de la chaîne - 1

Exemple :

```
>>> chaine = Maillon(21 ,Maillon(13,  Maillon ( 15, Maillon( 45))))
>>> print(chaine.maillon(2))
15
```

Ecrire le code de la méthode.

.....

.....

.....

.....

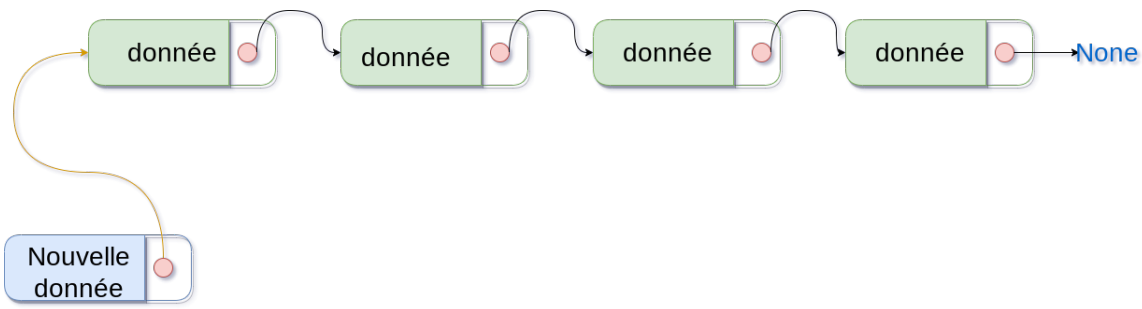
.....

.....

.....

Question 8 : Ajouter au début

On souhaite désormais pouvoir ajouter des maillons à nos instances de liste chaînée. Implémenter la méthode `ajouter_debut`, qui prendra en argument une valeur et qui l'ajoutera au début de la liste.



Exemple :

```
>>> m = Maillon( 1 , Maillon(2, Maillon(3, None )))
>>> m.ajouter_debut(0)
>>> print(m)
0 -> 1 -> 2 -> 3
```

Ecrire le code de la méthode.

.....

.....

.....

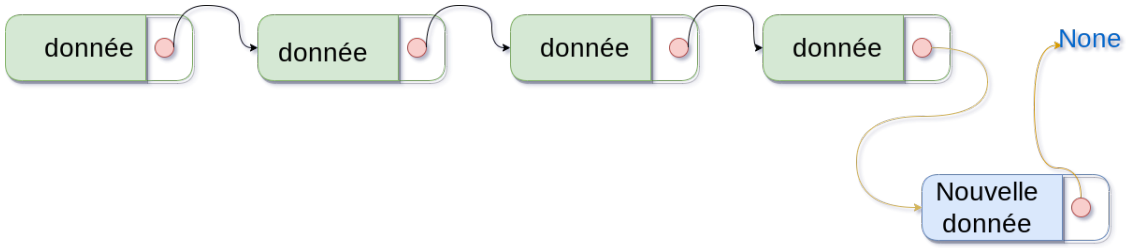
.....

.....

.....

Question 9 : Ajouter à la fin

Implémenter la méthode `ajouter_fin`, qui prendra en argument une valeur et qui l'ajoutera à la fin de la liste.



Exemple :

```
>>> m = Maillon( 1 , Maillon(2, Maillon(3, None )))
>>> m.ajouter_fin(4)
>>> print(m)
1 -> 2 -> 3 -> 4
```

Ecrire le code de la méthode.

. . . . . .

. . . . . .

. . . . . .

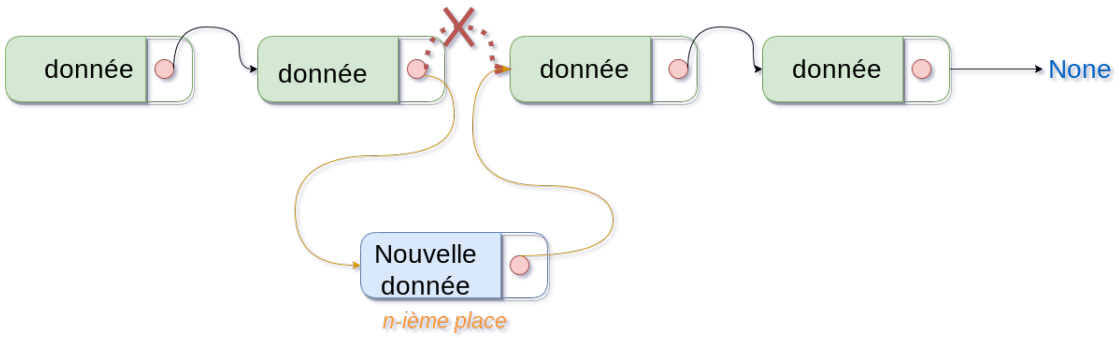
. . . . . .

. . . . . .

. . . . . .

Question 10 : Insérer à une position

Implémenter la méthode `inserer(valeur, n)` qui prendra en argument une valeur et une position `n` qui ajoutera un maillon ayant pour contenu valeur à l'index `n` (comme pour les tableaux on commencera à l'indice 0).



Exemple :

```
>>> m = Maillon( 1 , Maillon(2, Maillon(4, None )))
>>> m.inserer(3,2)
>>> print(m)
1 -> 2 -> 3 -> 4
```

Ecrire le code de la méthode.

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .