TP11 TNSI	Listes

I - La cellule

Celle-ci est déjà créée sous forme d'une classe dans le document *TDListesEleve.py.* Il s'agit de la même structure que dans le diaporama.

II - La liste

1 - Avec Python, créer une liste de 3 éléments (1, 2 et 3). Se référer au diaporama en cas de difficulté. *

2 - Avec Python, donner la syntaxe pour afficher chacun de ces éléments et la tester. *

III - Les actions courantes d'une liste

A - Longueur *

1 - Avec Python, écrire une fonction *longueur*, récursive, qui renvoie le nombre des éléments d'une liste. Une phase de réflexion sans PC peut être nécessaire afin d'identifier cas de base et cas récursif.

2 - Tester votre programme.

3 - Quelle est la complexité de cet algorithme ?

B - N-ième élément de la liste **

1 - Avec Python, écrire une fonction *n_ieme*, récursive, qui renvoie le N-ième élément de la liste passée en paramètre. Une phase de réflexion sans PC peut être nécessaire afin d'identifier cas de base et cas récursif.

2 - Modifier votre fonction afin que l'exception « indice invalide » soit levée si n est trop grand. Vous serez amené à utiliser le mot-clé *raise* pour cela.

3 - Tester votre programme.

4 - Quelle est la complexité de cet algorithme ?

IV - Une classe liste

1 - Créer une classe Liste qui possède un attribut *tete* initialisé à None. Cet attribut contiendra par la suite la première cellule de la liste. *

2 - Ajouter un élément à la fin de la liste

3 - En appelant les fonctions précédentes, créer les méthodes

- __getitem__(self,n), qui renvoie le $n^{ième}$ élément de la liste. *
- __len__(self) qui renvoie la longueur de la liste. *

4 - Tester ces méthodes

V - Fonctions supplémentaires à écrire en priorité

1 - Supprimer le ième élément de la liste **

2 - Ajouter un ième élément dans la liste **

3 - Concaténer deux listes ***

4 - Renverser une liste *** (la méthode récursive et la méthode non récursive peuvent être tentées. Laquelle est la plus rapide ?)