Soutien en algorithmique et programmation

Séance 9 : listes chainées circulaires

Introduction

On va travailler sur des listes chainées circulaires, ce qui signifie simplement que le suivant de la dernière cellule n'est pas `None` mais la première cellule. Les listes auront donc toutes un élément fictif en tête permanent (sentinelle). On utilisera la même classe `Cellule` que dans les séances précédentes et on implantera les fonctions ci-dessous dans un fichier `listes_circ.py`.

On pourra tester les fonctions écrites grace au programme principal suivant, à copier-coller dans le fichier `liste_circ.py`. La fonction `creer` alloue la sentinelle, la chaine à elle-même et renvoie une référence sur cette sentinelle en résultat.

```
def creer():
    """
      La sentinelle est chainee a elle-meme quand la liste est vide
    """
    fictif = Cellule('?', None)
    fictif.suiv = fictif
    return fictif

def main():
    """
      Programme principal
    """
    liste = creer()
    for _ in range(12):
        val = randint(0, 9)
        print("Insertion de", val, ":")
        inserer(liste, val)
        print("  ", end="")
        afficher(liste)
    for _ in range(5):
        val = randint(0, 9)
        print("Suppression de", val, ":")
        if supprimer(liste, val):
            print("  ", end="")
            afficher(liste)
        else:
            print("  valeur absente de la liste")
    print("Tri de la liste :")
    liste = trier_ins(liste)
    print("  ", end="")
    afficher(liste)
    print("Decoupage de la liste :")
    listes = decouper(liste)
    print("  ", end="")
    afficher(liste)
    for liste in listes:
        print("  ", end="")
        afficher(liste)
```

Affichage

Implanter une fonction `afficher(liste)` qui affiche le contenu de la liste passée en paramètre, sous le format suivant : `1 -> 2 -> 3 -> 4 -> FIN` par exemple.

Insertion à sa place

Implanter une fonction `inserer(liste, val)` qui insère la valeur passée en argument à sa place dans la liste **qu'on supposera triée par ordre croissant dans cette question**. Cette fonction ne renvoie rien car la liste est définie par une référence sur la sentinelle, qui ne changera pas.

Suppression de la première occurence

Implanter une fonction `supprimer(liste, val)` qui supprime la première occurrence de la valeur passée en argument dans la liste (et ne fait rien si la liste ne contient pas cette valeur). La méthode renvoie `True` ssi une cellule a bien été supprimée et `False` si la valeur n'était pas dans la liste.

Découpage en deux sous-listes

Implanter une fonction `decouper(liste)` qui renvoie deux listes circulaires composées des éléments de la liste initiale, répartis un sur deux dans chacune des sous-listes et dans le même ordre que dans la liste originale. Par exemple, si la liste initiale est `1 -> 2 -> 3 -> 4 -> 5 -> FIN`, les sous-listes renvoyées seront `1 -> 3 -> 5 -> FIN` et `2 -> 4 -> FIN`.

Tri par insertion

Implanter une fonction `trier_ins(liste)` qui trie la liste par ordre **décroissant** en utilisant l'algorithme du tri par insertion. Le principe de cet algorithme est le suivant :

- on détache l'élément de tête de la liste initiale et on parcours la liste résultat pour trouver l'endroit où insérer cet élément ;
- on insère l'élément à sa place dans la liste résultat ;
- le tri est terminé lorsque la liste initiale est vide.