TP 11 informatique

BCPST 1 2019-2020

Les tris

Points abordés

— Listes.
— Tri de listes.

---

1 Exercices

Exercice 1. On considère le code suivant :

```
def tri_bulle_non_fonctionnel(L) :
    for i in range(len(L)) :
        for j in range(i,len(L)) :
            if L[j+1]>L[j]:
                L[j],L[j+1] = L[j+1],L[j]
    return L
```

1. Trouver une liste L pour laquelle `tri_bulle_non_fonctionnel(L)` ne donne pas une liste triée dans l'ordre croissant.
2. Corriger le code pour obtenir un tri bulle fonctionnel.

Exercice 2.
1. Écrire une fonction `mini_place(L,i)` qui prend en argument une liste de nombres L et un entiers i et qui place la plus petite valeur de la liste $M = [L[i], L[i+1], \ldots]$ en position i.
2. En déduire une fonction `tri_selectif(L)` qui prend en argument une liste L et qui trie la liste L dans l'ordre croissant.

Exercice 3.
1. Écrire une fonction `a_sa_place(L,x)` qui prend en argument une liste de nombres L triée dans l'ordre croissant et un nombre x et qui insère la valeur x dans L de sorte que L reste triée dans l'ordre croissant.
2. En déduire une fonction `tri_insertion(L)` qui prend en argument une liste L et qui trie dans l'ordre croissant cette liste.

Exercice 4. La notion d'ordre n'est pas limitée au nombre : par exemple, il existe un ordre utilisé dans les dictionnaires appelé ordre lexicographique. Dans ce contexte, on considère qu'un mot m_1 est plus petit pour l'ordre lexicographique qu'un mot m_2 si le mot m_1 apparaît avant m_2 dans le dictionnaire. Par exemple, "cochon" est plus petit dans l'ordre lexicographique que "cochonnet". En résumé, l'ordre lexicographique est celui utilisé pour faire l'appel en classe.

1. Écrire une fonction `plus_petit_lex(chaine1,chaine2)` qui prend en argument deux chaînes de caractères et qui renvoit `True` si `chaine1` est plus petit que `chaine2` dans l'ordre lexicographique.
2. En vous inspirant de ce qui est fait pour les nombres écrire une fonction `tri_chaine(L)` qui prend en argument une liste de chaîne de caractères et qui trie celle-ci.

Exercice 5. Écrire une fonction qui prend en argument une liste L d'entiers compris entre 1 et 1000 et qui renvoit la liste triée dans l'ordre croissant. On effectuera au plus un seul parcours de la liste L.

Exercice 6. Écrire une fonction `recherche(L,mot)` qui prend en argument une liste de mots triés dans l'ordre lexicographique et qui renvoit -1 si le mot n'est pas dans L et i où i est la position du mot dans L sinon.

Exercice 7. Écrire une fonction `recherche(L,mot)` qui prend en argument une liste de mots triés dans l'ordre lexicographique et qui renvoit -1 si le mot n'est pas dans L et i où i est la position du mot dans L sinon.