Tris de listes

Le tri est une opération courante, et donc importante, en informatique. Trier une liste c'est ordonner ses éléments selon un ordre donné (numérique, alphabétique, lexicographique, etc.).
Précisons ici ce que nous entendons par liste : une liste est un ensemble d'éléments accessibles via un indice (allant de 0 à $n-1$ s'il y a n éléments) dont on peut changer les éléments (et en particulier échanger deux éléments de celle-ci). Les éléments de cette liste sont tous de même type, et on dispose d'une relation d'ordre qui permet de comparer deux éléments entre deux sous la forme « a < b ». Il existe de très nombreux algorithmes de tri, plus ou moins efficaces, en particulier selon les propriétés initiales de la liste à trier (mais nous ne traiterons pas de ces particularités).

Cet exercice est à faire en groupe.

1. Prendre les cartes numérotées (de 1 à 10) d'une même couleur d'un paquet de cartes, les mélanger et n'en conserver que cinq prises au hasard. Le principe est que l'on ne connaît pas l'ensemble des valeurs qui sont à trier. Placer ces cinq cartes faces cachées dans une file, c'est à dire les unes à côté des autres.

   Le but est de trouver un moyen, par déplacements, échanges, etc. des cartes de faire en sorte que les cartes soient à la fin triées par ordre croissant de numéro.
2. Avant de manipuler quoi que ce soit, se mettre d'accord dans le groupe, en discutant, sur la manière de procéder pour trier cette file.

   Attention à respecter la contrainte suivante : il n'est possible de voir (et comparer) que deux cartes en même temps. Autrement dit deux cartes doivent être replacées faces cachées une fois utilisées pour être comparées : on ne « voit » pas l'ensemble des numéros des cartes d'un seul coup d'œil.

   Il est cependant possible de retenir certaines valeurs, comme un indice dans la file ou la valeur d'une carte déjà vue, ces valeurs étant à voir comme étant enregistrées dans des variables (qui seront à nommer). On ne pourra cependant pas retenir l'ensemble des valeurs vues.
3. Une fois la stratégie de tri définie, l'appliquer pour en vérifier la validité. Corriger éventuellement les erreurs dans la stratégie, puis recommencer jusqu'à ce qu'une stratégie correcte soit trouvée.
4. Mettre à l'écrit la stratégie de tri utilisée, que nous appellerons maintenant algorithme de tri, le plus précisément possible, sachant qu'un autre groupe devra comprendre et appliquer cet algorithme à partir de ces explications.

   On précise à nouveau l'élément suivant pour homogénéiser les productions :

   On identifiera les cartes dans la file par des indices allant de 0 à 4 (pour 5 cartes).

   On pensera bien sûr à nommer les variables nécessaires.

   On nommera en particulier n la variable égale à la longueur de la liste à trier. L'algorithme doit fonctionner quelle que soit le nombre d'éléments à trier.
5. Donner votre algorithme de tri à un groupe, prendre un algorithme d'un autre groupe (si le nombre de groupes est pair, vous pouvez vous contenter d'échanger votre algorithme avec un autre groupe). Appliquer alors cet algorithme, le plus « bêtement » possible, c'est à dire en cherchant d'éventuelles failles qui feraient échouer le tri.

6. Dans le cas où l'algorithme n'est pas assez précis ou n'est pas correct, indiquer à l'autre groupe l'erreur ou les erreurs détectées.

   Proposer éventuellement des indications de correction au groupe qui récupérera son algorithme.

7. Récupérer l'algorithme de votre groupe et, si nécessaire, l'améliorer, éventuellement en communiquant avec le groupe qui l'a reçu.
8. Une fois l'algorithme de tri fonctionnel, récupérer le fichier Tris.py disponible sur Moodle, puis compléter la définition de la fonction `tri` :
9. Le code du fichier Tris.py contient des fonctions permettant de tester un algorithme de tri. Nous avons déjà vu cela dans l'activité sur les tris.

   Exécuter alors la fonction de test sur la fonction de tri définie précédemment.

10. Deux algorithmes sont à connaître en première NSI, le tri par sélection et le tri par insertion, qui sont des tris assez « naturels ». Ceux-ci sont donnés dans le cours, et seront étudiés en détail suite à cette activité.

    Ouvrir le cours contenant ces algorithmes de tri, les traduire sous forme de fonctions Python et exécuter la fonction de test sur ces fonctions.

11. Observer les différences de temps d'exécution moyens entre les différents algorithmes de tri.

    Pour cela, dé-commenter la partie correspondante dans le fichier Tris.py.

    Le code utilise le module Time

12. Observer les graphiques comparatifs des temps d'exécution.

    Pour cela, dé-commenter la partie correspondante dans le fichier Tris.py.

    Le code utilise le module matplotlib

13. Question subsidiaire : commenter les codes donnés dans les deux questions précédentes pour les expliquer.

Note : En terminale NSI on étudie un tri plus efficace que les deux algorithmes vus en première, à savoir le tri fusion.