NSI 1ère - Algorithmique - Tris 4

QK

Plan

TD programmer tri par insertion

1. reprendre l'algorithme, le programmer en python
2. programmer version qui compte les échanges
3. mesure pratique des durées, comparaison avec tri rapide (à taper) et timsort
4. si le temps faire prog tri à bulle
5. montrer que tri rapide et timsort $O(n \log n)$

Tri par insertion, travaux dirigés.

1. Programmer le tri par insertion

En partant de l'algorithme ci-dessous, écrire le programme Python correspondant.

On prendra soin de créer une fonction pour réaliser les échanges.

```
Tri Insertion(tableau t, entier n)
i = 0
Tant que i < n
    j = i
    Tant que j > 0 et t[j-1] > t[j]
        echanger t[j] et t[j-1]
        j = j - 1
    fin tant que
    i = i + 1
fin tant que
```

2. Seconde version : décompte des échanges.

1. Quel est le pire des cas en terme de nombre d'échanges pour le tri par insertion ?
2. Créer une fonction qui génère un tableau trié par ordre décroissant.
3. Créer une seconde version du tri qui renvoie un entier : le nombre d'échanges réalisés par le tri.
4. Construire deux listes Python, une liste de tailles croissantes et une liste du nombre d'échanges pour le tri par insertion d'un tableau trié de manière décroissante.
5. À l'aide de `pylab` représenter la courbe du nombre d'échanges en fonction de la taille.
6. Pour les tailles précédentes, construire la liste des durées écoulées.
7. Toujours à l'aide de `pylab` représenter la durée en fonction de la taille.

3. Complément : d'autres tris.

timsort.

timsort est le nom du tri utilisé par Python (méthode `sort` et fonction `sorted`)

Lire la documentation du tri natif `help(list.sort)` et reprendre la figure précédente pour le tri natif.

On construira deux courbes : le tri natif seul et les deux tris ensemble.

Tri à bulle

1. En partant de l'article wikipédia sur le tri à bulle programmer une fonction `tri_a_bulle` en python.

2. Ajouter à vos courbes précédentes celle du tri à bulle.

projets complémentaires

Illustrer les tris.

Construire une animation en python illustrant les tris. On pourra, par exemple, utiliser des bâtons de taille croissante et montrer leur position à des moments définis de l'algorithme.

On peut utiliser des couleurs pour mettre en avant l'élément couramment trié.