Exemples d’algorithmes de recherche dans un tableau ou une liste.

Le problème de la recherche

Collections ordonnées d’objets : les tableaux et les listes

suites indexées d’éléments
Les tableaux :
- zone mémoire contiguë,
- accès à un élément d’indice i en temps constant,
- utilisation en scratch et en python (type list)

Les listes chainées :
- mémoire non contiguë
- maillon relié par référence, pointeur
- accès à l’élément d’indice i en temps O(i)/ proportionnel à i
- exemple d’implémentation en python

description du problème

Etant donnée une collection indexés d’objets L et un objet o, déterminer si o appartient à L.
Variantes :
- la position
- nombre d’occurence
- min/max

L est un tableau ou une liste

---

comparaison de la complexité , accès aux éléments, entre tableau et liste

Recherche dans une structure non triée

Éléments dans le désordre ou non comparable

1- Recherche séquentielle, l’algorithme de base

```
Description en pseudo-code de l'algo
Complexité en O(len(L)) pour les listes ou les tableaux
exemple de situation
implémentation (récursive ou itérative en python et/ou en scratch )

autres recherches:
- recherche de position
- recherche à la position
- recherche de doublons
- nombre d’occurrence
complexité dans les deux cas
```

2- Recherche séquentielle : algorithmes particuliers

```
recherche auto-adaptative ( application, pourquoi utiliser cet algo, plus
efficace sur les listes que sur les tableaux)

recherche dans un tableau multi-dimensionnelle (boucles imbriqués )
```

Recherche dans une structure triée

contexte : éléments comparables 2 à 2
collection triée selon une relation d’ordre totale.

1- Recherche séquentielle

```
cas d'arrêt si l'élément rechercher est plus grand que T[i]
recherche de position : complexité moindre
recherche à la position : même complexité
recherche de doublon : complexité moindre
nombre de d’occurrence : complexité moindre
```

2- Recherche par dichotomie

```
activité introductive :jeu de la devinette
observation : si L(i)<o, on peut ignorer les indices <= i, et
réciproquement
principe
implémentation sur les tableaux (itératif,récursif)
complexité sur les tableaux
sur les listes : inefficaces ( pas d'accès direct aux éléments)
```

Ouverture : Recherche par interpolation

```
principe
lien avec le dictionnaire (au sens littéral)
pseudo-code
avantage, inconvénient
```