STRUCTURES DE DONNÉES ET ALGORITHMES · CHAPITRE 12

Chercher sans supposer les données triées

Tracez une recherche séquentielle, définissez le résultat en cas d'absence et précisez le comportement avec des doublons.

Par Ali ZAINOUL · Cours gratuit

Ce que vous allez apprendre

- Retrouver la première occurrence sans supposer un ordre préalable.
- La recherche linéaire ne suppose pas une entrée triée.
- Le contrat présenté renvoie la première occurrence, ou une sentinelle d'absence.

Prérequis. Avoir étudié les chapitres DSA précédents ; connaître tableaux, invariants et récursivité.

Partir d'un besoin

Pour retrouver 7 dans [4, 7, 2, 7], aucune règle d'ordre ne permet d'éliminer une moitié de la suite. Il faut examiner les valeurs jusqu'à trouver une occurrence ou atteindre la fin.

Définir la structure et les conventions

Une **recherche linéaire** compare les éléments dans l'ordre. Le contrat de notre fonction renvoie la position de la première occurrence de la cible. Si elle est absente, Python renvoie -1 et MATLAB 0. Ces sentinelles ne doivent pas servir à relire la collection.

Avant chaque comparaison, l'invariant est : aucune position déjà examinée ne contient la cible. Si la valeur courante convient, on a donc trouvé la première occurrence. Si le parcours se termine, toutes les positions ont été exclues. Une suite vide mène directement au résultat d'absence.

Suivre les états

Recherche de 7

Position Python / MATLAB	Valeur	Décision
0 / 1	4	Continuer
1 / 2	7	Renvoyer cette position

Comparer les réalisations

Python affiche 1 et MATLAB 2 : c’est la même deuxième position. Le script MATLAB linear_search_demo.m contient la fonction locale à sa fin. Comparer les indices bruts entre les langages sans leur convention donnerait une fausse différence.

Pseudo-code

```
// example.pseudo
// linear_search
FOR index FROM 0 TO length(values) - 1 INCLUSIVE DO
    IF values[index] == target THEN
        RETURN index
    END IF
END FOR
RETURN -1
```

Python

```python
# linear_search.py
def linear_search(values, target):
    for index in range(len(values)):
        if values[index] == target:
            return index
    return -1
values = [4, 7, 2, 7]
position = linear_search(values, 7)
print(position)
```

MATLAB

```matlab
% linear_search_demo.m
values = [4, 7, 2, 7];
position = linear_search(values, 7);
disp(position);
function position = linear_search(values, target)
    position = 0;
    for index = 1:numel(values)
        if values(index) == target
            position = index;
            return;
        end
    end
end
```

Justifier et compter

Au meilleur cas non vide, une comparaison suffit : Θ(1). Au pire, n comparaisons donnent Θ(n). La mémoire auxiliaire est Θ(1). Une moyenne n/2 ne peut pas être annoncée sans hypothèses sur la présence et la distribution des positions de la cible.

Prévoir puis expérimenter

Testez la cible au début, à la fin, absente, puis une collection vide. Avec des doublons, vérifiez que la première occurrence est renvoyée. Ne lisez jamais values[position] avant d'avoir exclu la sentinelle.

Exercice corrigé

▼ Peut-on interrompre la recherche dès qu'une valeur dépasse la cible ?

Seulement avec une précondition d'ordre adaptée. Dans [9, 7], dépasser 7 au premier élément ne prouve pas son absence.

À retenir

- La recherche linéaire ne suppose pas une entrée triée.
- Le contrat présenté renvoie la première occurrence, ou une sentinelle d'absence.
- Ne réutilisez jamais la sentinelle comme un indice de collection.

Version web : https://www.clearcode.fr/courses/programming/dsa/search-sort/linear-search