R. Malgouyres, R. Zrour et F. Feschet
Initiation à lalgorithmique et à la programmation en C,
Cours avec 129 exercices corrigés, 3e édition,
DUNOD, Collection Sciences Sup, 2014,
Nouvelle présentation 2015.

Algorithmique et programmation en C

TP n° 4
Recherche dichotomique dans un tableau trié

Objectifs :
Le but du TP est d'étudier la recherche dichotomique dans un tableau trié.

Note : On mettra les solutions de tous les exercices dans le même fichier.

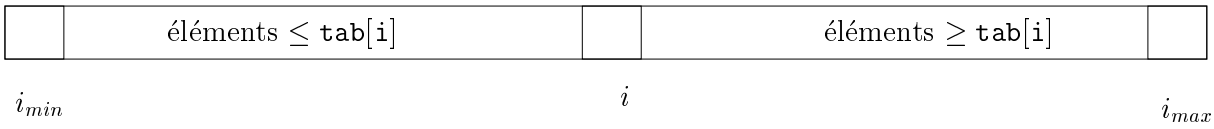
On cherche à savoir si un tableau de n nombres entiers contient une certain nombre a. On suppose que le tableau est trié dans l'ordre croissant, c'est à dire que chaque élément est inférieur ou égal au suivant dans le tableau.

On pourrait évidemment tester tous les nombres du tableau en les comparant à a, mais cela nécessiterait (dans le pire des cas) n comparaisons. On se propose d'appliquer une méthode qui nécessite moins d'opération. La méthode s'appelle la recherche dichotomique.

À chaque étape, on recherche un élément égal à a entre deux indices i_{min} et i_{max}. Au départ, l'élément peut se trouver n'importe où (entre $i_{min} = 0$ et $i_{max} = n - 1$)

On compare l'élément d'indice $i = (i_{min} + i_{max})/2$ avec a. Trois cas peuvent se produire :

- Si l'élément d'indice i est égal à a, l'algorithme se termine : le nombre a se trouve bien dans letableau.
- Si l'élément d'indice i est supérieur à a, un élément égal à a ne peut se trouver qu'à un indice plus petit que i, puisque le tableau est trié. Le nouvel i_{max} vaut $i - 1$.
- Si l'élément d'indice i est inférieur à a, un élément égal à a ne peut se trouver qu'à un indice plus grand que i. Le nouvel i_{min} vaut $i + 1$.



Á chaque étape, un élément égal à a ne peut se trouver qu'entre i_{min} et i_{max}. On itère ce procédé jusqu'à ce que i_{min} devienne strictement supérieur à i_{max}. Il ne reste plus qu'à tester un seul élément.

Comme à chaque étape le nombre d'éléments est divisé par 2, le nombre k d'étapes est tel que $\frac{n}{2^k} \geq 1$. Donc $k \leq \log_2(n)$.

Exercice 1 Écrire une fonction qui lit au clavier un tableau d'entiers. On lira le nombre d'éléments au clavier que les éléments rentrent bien dans l'espace mémoire alloué.

Exercice 2 Écrire une fonction de recherche dichotomique d'un élément dans un tableau trié. La fonction prend en paramètre un tableau, son nombre d'éléments, et un élément a à rechercher dans le tableau. La fonction renvoie 1 l'élément est présent et 0 sinon. On fera un `main` qui teste cette fonction

Exercice 3 Écrire une fonction qui détermine si deux tableaux on une intersection vide.

Exercice 4 Écrire une fonction qui détermine si tous les éléments d'un tableau sont présents dans un autre tableau.