

Algorithmique procédurale

TD de Révision

1) Tableaux

a) Deux tableaux sont *égaux* s'ils contiennent les mêmes éléments aux mêmes positions. Deux tableaux sont *similaires* s'ils contiennent les mêmes éléments mais pas forcément aux mêmes positions. Par exemple: [9,6,7,6,3,6,1,9] et [1,3,6,6,6,7,9,9] sont similaires alors que [9,6,7,6,3,6,1,9] et [9,9,7,6,3,6,1,9] ne le sont pas. Deux tableaux sont *comparables* si l'ensemble des valeurs qu'ils contiennent est le même. Par exemple, les tableaux [1,3,6,7,9],[6,7,3,6,1,9] et [9,6,7,6,3,6,1,9] sont tous les trois comparables.

- Écrire une méthode qui teste si deux tableaux sont égaux.
- Écrire une méthode qui teste si deux tableaux sont similaires.
- Écrire une méthode qui teste si deux tableaux sont comparables.

b) Un tableau carré à deux dimensions contient des lettres à raison d'un caractère par case du tableau. Ecrire une fonction qui détermine si un mot donné est présent dans le tableau en ligne ou en colonne. Ce mot est stocké dans un tableau à une dimension à raison d'un caractère par case.

2) Tris

a) Quels tris connaissez-vous et quel est leur principe?

Appliquez-les au tableau d'entiers suivant : [4, 8, 2, 10, 1, 9, 7, 6, 3, 5]

b) Ecrire un algorithme qui trie un tableau de booléens de sorte que tous les FAUX se trouvent à gauche du tableau, et tous les VRAI à droite du tableau. Attention, votre algorithme doit avoir la complexité $O(n)$, et non pas $O(n \cdot \log n)$ voire $O(n^2)$. Une solution basée sur le tri casier ne compte pas.

c) « Tri drapeau » (difficile) : un tableau contient des éléments rouges, blancs et bleus. Triez ce tableau pour que les éléments rouges soient au début, les blancs au milieu et les bleus à la fin. De nouveau, l'algorithme doit être $O(n)$, c'est-à-dire qu'il faut ne tester qu'une fois la couleur d'un élément. Une solution basée sur le tri casier ne compte pas.

3) Pointeurs C

Explicititez ce qu'il se passe à chaque ligne du « main » dans le code suivant:

```
#include <stdio.h>
void foo(int val, int res) {
    res = val;
}
void bar(int val, int *res) {
    *res = val;
    if (val < 0) *res *= -1;
}
int main() {
    int a, b, c, d;
    int * p1, * p2, * p3, * p4;
    a = 7;
    b = -3;
    c = 12;
    d = -5;
    p1 = &c;
    p2 = &a;
    p3 = &b;
    p4 = p1;
    foo(c, *p3);
    *p1 += a++;
    *p3 += ++(*p2);
    bar(d, p1);
    printf("Résultat: %d,%d,%d,%d,%d,%d,%d,%d \n", a, b, c, d, *p1, *p2, *p3, *p4);
    return 0; // ou return EXIT_SUCCESS
}
```