Exercice 1 : Dichotomie

1. Rappeler le programme impératif de la recherche dichotomique.
2. Écrire une version récursive de cet algorithme. Il sera peut-être nécessaire d'ajouter des arguments.
3. Créer une liste l de cinquante éléments triés.
4. Tester les deux fonctions sur la liste l.
5. Rappeler la complexité de l'algorithme de recherche dichotomique.

Exercice 2 : Le tri rapide est un autre exemple d'algorithme utilisant la méthode *diviser pour régner*. C'est un algorithme naturellement récursif que nous pouvons décrire ainsi :

— Choisir un élément pivot.

— Sélectionner tous les éléments inférieurs au pivot.

— Sélectionner tous les éléments supérieurs ou égaux au pivot.

— Placer récursivement à gauche du pivot les éléments inférieurs à ce-dernier et à droite les éléments supérieurs.

1. Écrire une fonction **tri_rapide(tab : list) list** qui implémente l'algorithme du tri rapide. Nous choisirons le premier élément du tableau comme pivot.
2. Construire en compréhension une liste l de vingt éléments compris entre 0 et 100.
3. Tester la fonction de tri sur la liste l.

Remarque : Le tri rapide a une complexité en $O(n \times log_2(n))$.

Exercice 3 : La fonction *mystere* implémente un algorithme du type *diviser pour régner*.

```
def mystere(tab: list, debut: int = 0, fin: int = -1)->int:
    #initialise 'fin'
    if fin == -1:
        fin = len(tab)
    if debut == (fin-1):
        return tab[debut]
    else:
        milieu = (debut + fin)//2
        gauche = mystere(tab, debut, milieu)
        droite = mystere(tab, milieu, fin)
        if (gauche > droite):
            return gauche
        else:
            return droite
```

Soit la liste :

```
tab = [5, 71, 23, 45, 28, 89, 63, 39]
```

1. Dessiner l'arbre des séparations engendré par la fonction sur la liste *tab*.
2. Dessiner l'arbre des recombinaisons. Quelle valeur renvoie l'appel **mystere(tab)** ?
3. Que fait cette fonction ?

Christophe Viroulaud