Exercice: Recherche par dichotomie

On redonne l'algorithme de recheche par dichotomie en langage python.

```
a = 0
b = len(t) - 1
while a <= b :
    m = (a + b)//2
    if v < t[m] :
        b = m - 1
    elif v > t[m] :
        a = m + 1
    sinon return m
return -1
```

La valeur -1 est retournée lorsque le nombre n'est pas présent dans le tableau. On pourrait choisir une autre valeur ou même une chaine de caractère !

Exercice 1

Max et Lilly joue à devine nombre. Max pense à un nombre entier compris entre 1 et 100 et Lilly doit le trouver. À chaque nombre proposé par Lilly, Max lui dit s'il est plus petit ou plus grand.
Si Lilly applique la recherche par dichotomie, en combien d'essais au maximum peut-elle trouver le nombre de Max ?

Exercice 2

Soit T un tableau trié tel que : $T = [8, 8, 17, 21, 23, 27, 28, 45, 57, 71, 77, 84, 88, 95, 97]$.

1. On veut vérifier la présence du nombre 23 dans le tableau.
   - **a** Retracer toutes les étapes de l'algorithme de recherche par dichotomie.
   - **b** Combien d'itérations ont été nécessaires ?
2. Faire de même avec le nombre 88.
3. Faire de même avec le nombre 75.

Exercice 3

1. Créer un tableau T de valeurs choisies aléatoirement entre 1 et 1000.
2. Trier ce tableau avec une des fonctions de tri de python.
3. Écrire l'algorithme de recherche par dichotomie en langage python.
4. Tester votre programme avec la recherche de différentes valeurs du tableau et des valeurs qui ne sont pas dans le tableau.

Exercice 1

La situation nécessitant le maximum d'essais est celle qui conduit à avoir $a = b$. On peut chercher le nombre d'itérations pour avoir $a = b = 1$. Rassemblons les valeurs dans un tableau :

itérations		1	2	3	4	5	6	7
$m = (a+b)//2$		50	25	12	6	3	1	1
a	1	1	1	1	1	1	1	1
b	100	49	24	11	5	2	1	0

Ce tableau montre qu'au maximum 7 itérations seront nécessaires pour trouver le nombre pensé par Max.

Exercice 2

Soit T un tableau trié tel que : $T = [8, 8, 17, 21, 23, 27, 28, 45, 57, 71, 77, 84, 88, 95, 97]$.

1 On veut vérifier la présence du nombre 23 dans le tableau.

a Le tableau trié contient 15 valeurs donc $a = 0$ et $b = 14$.

itérations		1	2	3	4
$m = (a+b)//2$		7	3	5	4
$T[m]$		45	21	27	23
a	0	0	4	4	
b	14	6	6	5	

b 4 itérations ont été nécessaires pour touver 23.

2 Faire de même avec le nombre 88.

a Le tableau trié contient 15 valeurs donc $a = 0$ et $b = 14$.

itérations		1	2	3	4
$m = (a+b)//2$		7	11	13	12
$T[m]$		45	84	95	88
a	0	8	12	12	
b	14	14	14	12	

b 4 itérations ont été nécessaires pour touver 88.

3 Faire de même avec le nombre 75.

a Le tableau trié contient 15 valeurs donc $a = 0$ et $b = 14$.

itérations		1	2	3	4
$m = (a+b)//2$		7	11	9	10
$T[m]$		45	84	71	77
a	0	8	8	10	10
b	14	14	10	10	9

$a > b$ donc la valeur n'est pas dans le tableau.

b 4 itérations ont été nécessaires pour montrer que 75 n'est pas dans le tableau..

Exercice 3

1 Tableau T de valeurs choisies aléatoirement entre 1 et 1000. On peut créer une fonction **creer_tab_alea** qui renvoie un tableau aléatoirement rempli.

2 Trier ce tableau avec une des fonctions de tri de python. On utilise la méthode de tri **sort** de python pour trier le tableau

```
import random as r

def creer_tab_alea(n):
    """Crée un tableau de dimension n dont les valeurs sont des entiers choisis aléatoirement entre 1 et 1000"""
    return [r.randint(1,100) for i in range(n)]

t=creer_tab_alea(20)
t.sort()
print(t)
```

```
[8, 11, 13, 22, 29, 31, 34, 37, 38, 48, 63, 69, 69, 78, 85, 89, 92, 94, 94, 98]
```

3 Algorithme de recherche par dichotomie en langage python.

```
def dichotomie(t,v):
    a=0
    b=len(t)-1 # indice du dernier nombre du tableau
    i=0 # pour compter le nombre d'itérations
    while a<=b:
        i=i+1 # on ajoute 1 à chaque tour de la boucle while (tant que)
        m=(a+b)//2
        if v<t[m]:
            b=m-1
        elif v>t[m]:
            a=m+1
        else:
            return m,i # renvoie la position du nombre trouvé et le nombre d'itérations de la boucle
    return None,i # # renvoie aucun nombre trouvé et le nombre d'itérations de la boucle
```

4 Tester votre programme avec la recherche de différentes valeurs du tableau et des valeurs qui ne sont pas dans le tableau.

a On cherche le nombre 29 présent dans le tableau : $t[4] = 29$; 2 itérations suffisent !

```
dichotomie(t,29)
```

```
(4, 2)
```

b On cherche le nombre 39 qui n'est pas dans le tableau ; 5 itérations nécessaires.

```
dichotomie(t,39)
```

```
(None, 5)
```

a Importer le module **math** et calculer la valeur $\log_2(n)$:

```
import math
Imax=math.log2(20)
print("Il faut au maximum",int(Imax)+1,"itérations pour trouver un nombre ou affirmer qu'il n'y est pas !")
```

```
Il faut au maximum 5 itérations pour trouver un nombre ou affirmer qu'il n'yest pas !
```

b Le nombre maximal d'itérations pour un tableau de 20 valeurs est 5.

c Boucle qui donne toutes les recherches dichotomiques pour trouver tous les nombres de votre tableau et le nombre d'itérations dans chaque cas. On rappelle le tableau utilisé dans l'exercice :

```
t=[8, 11, 13, 22, 29, 31, 34, 37, 38, 48, 63, 69, 69, 78, 85, 89, 92, 94, 94, 98]
```

```
N=int(Imax)+1
for k in range(N):
    liste=[]
    for v in t:
        res=dichotomie(t,v)
        if res[1]==k+1:
            liste.append(v)
    print(liste,"en",k+1,"itération(s)")
```

```
[48] en 1 itération(s)
[29, 85] en 2 itération(s)
[11, 34, 69, 69, 94, 94] en 3 itération(s)
[8, 13, 31, 37, 63, 89] en 4 itération(s)
[22, 38, 78, 92, 98] en 5 itération(s)
```