Deux exemples de recherche dichotomique

1. On considère une première fonction de recherche d'un entier `val` dans un tableau d'entiers `tab` trié dans l'ordre croissant :

```
def recherche_dicho(tab, val):
    gauche = 0
    droite = len(tab) - 1
    while droite - gauche > 0:
        milieu = (gauche + droite) // 2
        print(gauche, milieu, droite)
        if tab[milieu] < val:
            gauche = milieu + 1
        elif tab[milieu] > val:
            droite = milieu - 1
        else:
            gauche = milieu
            droite = milieu
        print(gauche, milieu, droite)
    if gauche == droite:
        return tab[gauche] == val
    else:
        return False
```

Propriété 1

Dans cette première version, la boucle maintient **l'invariant** :

`gauche` $\leqslant$ position de `val` $\leqslant$ `droite` (si `val` est dans le tableau).

La zone de recherche [`gauche` ; `droite`] est un intervalle fermé qui est divisé environ par deux à chaque itération et la boucle s'arrête lorsque la zone de recherche est vide ou réduite à un élément (quand `droite` $\leqslant$ `gauche`).

En sortie de boucle, `val` est dans le tableau si et seulement si la zone de recherche est réduite à un seul élément qui est égal à `val`.

On donne quatre exemples d'évaluation :

```
recherche_dicho([-4, 5, 7, 9, 12, 14, 18, 21], -4)
Out[2]: True

recherche_dicho([-4, 5, 7, 9, 12, 14, 18, 21], 10)
Out[3]: False

recherche_dicho([-4, 5, 7, 9, 12, 14, 18, 21], 5)
Out[4]: True
```

On donne les premières lignes d'un tableau d'état des variables `gauche`, `milieu` et `droite` au début et à la fin de chaque itération de boucle de la fonction `recherche_dicho`.

	`gauche`	`milieu`	`droite`
Itération 1 début (ligne 6)	...	...	...
Itération 1 début (ligne 14)	...	...	...
Itération ...début (ligne 6)	...	...	...
Itération ...début (ligne 14)	...	...	...

Question

Recopier et compléter le tableau d'état des variables pour chacune des quatre évaluations suivantes. Préciser dans chaque cas la valeur renvoyée par la fonction.

- `recherche_dicho([-4, 5, 7, 9, 12, 14, 18, 21], 10)`
- `recherche_dicho([-4, 5, 7, 9, 12, 14, 18, 21], 5)`
- `recherche_dicho([-4, 5, 7, 9, 12, 14, 18, 21], 12)`
- `recherche_dicho([-4, 5, 7, 9, 12, 14, 18, 21], 22)`

2. Refaire un tableau d'état des variables pour les mêmes quatre évaluations mais avec la fonction `recherche_dicho2` et les valeurs des variables `gauche, droite, milieu` aux lignes 6 et 11 :

```
1  def recherche_dicho2(tab, val):
2      gauche = 0
3      droite = len(tab)
4      while droite - gauche > 1:
5          milieu = (gauche + droite) // 2
6          print(gauche, milieu, droite)
7          if tab[milieu] <= val:
8              gauche = milieu
9          else:
10             droite = milieu
11         print(gauche, milieu, droite)
12     return tab[gauche] == val
```

Propriété 2

Dans cette seconde version, la boucle maintient **l'invariant** :

`gauche` $\leqslant$ position de `val` $<$ `droite` (si `val` est dans le tableau).

La zone de recherche [`gauche` ; `droite`[est un intervalle semi-ouvert à droite qui est divisé environ par deux à chaque itération et la boucle s'arrête lorsque la zone de recherche est réduite à un seul élément (quand `droite` = `gauche` + 1 car l'intervalle est semi-ouvert à droite).

En sortie de boucle, `val` est dans le tableau si et seulement s'il est égal à `tab[gauche]`.