TD n° 2 : rappels sur la récursivité

Exercice 1. *Calcul de puissances*
On souhaite calculer la puissance (d'exposant entier positif) d'un nombre en n'utilisant que des multiplications (et donc pas l'opérateur `**`).

1. Écrire une fonction *non* récursive d'entête
   ```
   def puissance_ite(x: float, n:int) -> float:
   ```
   qui renvoie la valeur de x^n en utilisant le fait que $x^n = \underbrace{x \times x \times \cdots \times x}_{n \text{ fois}}$.
2. On observe que, pour tout $x \in \mathbb{R}$ et $n \in \mathbb{N}$,
   $$x^{n+1} = x \times x^n.$$
   a) Écrire une fonction récursive d'entête
   ```
   def puissance(x: float, n: int) -> float:
   ```
   qui renvoie la valeur de x^n basée sur l'observation ci-dessus.
   b) Démontrer la terminaison de cette fonction.
   c) Estimer le nombre $C(n)$ de multiplications effectuées lors de l'appel `puissance(x, n)`.
3. On observe maintenant que, pour tout $x \in \mathbb{R}$ et $n \in \mathbb{N}$,
   $$x^n = \begin{cases} x^{\frac{n}{2}} \times x^{\frac{n}{2}} & \text{si } n \text{ est pair,} \\ x \times x^{\frac{n-1}{2}} \times x^{\frac{n-1}{2}} & \text{si } n \text{ est impair.} \end{cases}$$
   a) Écrire une fonction récursive d'entête
   ```
   puissance_rapide(x: float, n: int) -> float:
   ```
   qui renvoie la valeur de x^n basée sur cette nouvelle observation.
   b) Estimer le nombre $D(n)$ de multiplications effectuées lors de l'appel `puissance_rapide(x, n)`. On pourra se contenter du cas où $n = 2^p$ avec $p \in \mathbb{N}$.

Exercice 2. *Palindromes*
Un palindrome est un mot qui se lit indifféremment de gauche à droite ou de droite à gauche comme ICI, KAYAK ou RESSASSER.
Donner deux versions, l'une récursive, l'autre non, d'une fonction d'entête
```
def est_palindrome(mot: str) -> bool:
```
qui renvoie `True` si `mot` est un palindrome et `False` sinon.

⇝ *On pourra aussi faire l'épreuve 3.01 (chaîne de caractères palindrome) du challenge.*

Exercice 3. *Dichotomie*
On suppose que l'on dispose d'une liste de flottants `tab` triée par ordre croissant. On veut déterminer si un flottant `x` appartient ou non à cette liste. On souhaite tirer partie que la liste soit triée en procédant par dichotomie.

1. Rappeler la complexité de la recherche dichotomique d'une valeur dans une liste de longueur n.
2. Écrire une fonction récursive d'entête
   ```
   def dicho(x: float, tab: [float], deb: int, fin: int) -> bool:
   ```
   qui renvoie `True` ou `False` suivant que `x` se trouve ou non dans la liste `tab` entre les indices `deb` (inclus) et `fin` (exclu).

Exercice 4. *Tri fusion*

Le tri fusion repose sur le principe « diviser pour régner » qui consiste à :

1. diviser le problème initial en sous-problèmes, ici on coupe la liste à trier en deux sous-listes plus petites ;
2. traiter les sous-problèmes, ici on trie chaque sous-liste en revenant au premier point si elle contient au moins deux éléments ;
3. combiner les résultats obtenus pour obtenir une solution au problème initial, ici on fusionne les sous-listes triées.

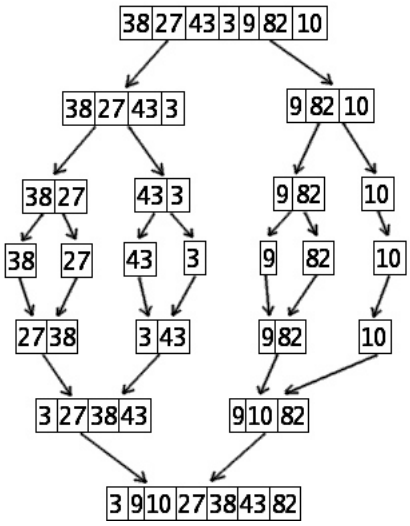


FIGURE 1 – Illustration du tri fusion

1. Écrire une fonction `fusion` d'entête

```
def fusion(tab1: list, tab2: list) -> list:
```

qui prend en arguments deux listes triées `tab1` et `tab2` et qui renvoie une nouvelle liste triée contenant tous les éléments de ces deux listes.

Par exemple, `fusion([1,2,4,6], [3,5,6,6,8])` doit renvoyer `[1,2,3,4,5,6,6,6,8]`.

2. Écrire une fonction d'entête

```
def tri_fusion(tab: list) -> list:
```

qui renvoie une nouvelle liste qui est une version triée de celle donnée en argument. On procèdera bien sûr via le procédé récursif défini ci-dessus.

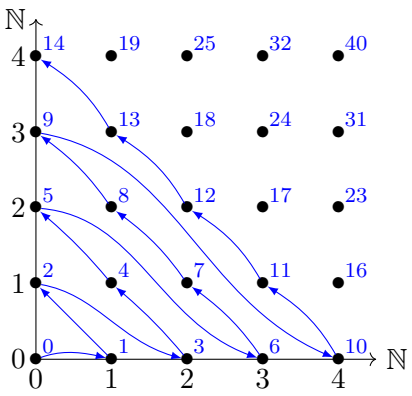
Exercice 5. ★ *Numéroter les éléments de* $\mathbb{N}^2$

Il est possible de numéroter chaque élément $(i, j) \in \mathbb{N} \times \mathbb{N}$ selon le procédé suggéré ci-contre : $(0, 0)$ est numéroté par 0 puis $(1, 0)$ par 1, $(0, 1)$ par 2, $(0, 2)$ par 3, $(1, 1)$ par 4, etc.

Écrire une fonction récursive d'entête

```
def numero(i: int, j: int) -> int:
```

qui renvoie le numéro du point de coordonnées (i, j) selon ce principe de numérotation.



⇝ Faire l'épreuve 5.05 (*Remplissage d'image*) du challenge.