Exercice 1 :[Question barrière]

Compléter le script **Python** suivant (dont on donne la première ligne) pour qu'il calcule et affiche le premier entier $n \in \mathbb{N}$ tel que

$$\left(\frac{1}{\sqrt{\mathrm{e}}}\right)^n \leqslant 10^{-4}$$

```
1 import numpy as np
```

Démonstration.

```
1 import numpy as np
2 n= 0
3 while (1 / np.sqrt(np.e))**n > 10**(-4):
4     n = n + 1
5 print(n)
```

ou la version plus *fancy* :

```
1 import numpy as np
2 n= 0
3 while (1 / np.sqrt(np.e))**n > 10**(-4):
4     n = n + 1
5 print(f'La valeur de n recherchée est {n}')
```

□

Exercice 2 : On considère la suite $(u_n)_{n\in\mathbb{N}}$ définie par : $\begin{cases} u_0 = 0 \\ \forall n \in \mathbb{N},\ u_{n+1} = \dfrac{1+u_n}{1+\mathrm{e}^{u_n}} \end{cases}$.

On admet que, pour tout $n \in \mathbb{N}^*$, $|u_n - \alpha| \leqslant \frac{1}{2}M^{n-1}$ où $M = 0,125$ et $\alpha = \lim\limits_{n\to+\infty} u_n$.

Compléter la fonction **Python** suivante pour qu'elle

- prenne en argument un réel `epsilon` > 0.
- renvoie une valeur approchée de α à `epsilon` près.

```
1 import numpy as np
2 def valeur_approchee(epsilon):
3     n = _____
4     u = _________
5     while ____________________________________
6         n = n + 1
7         u = ________________________
8     return _____
```

Démonstration.

```
import numpy as np
def valeur_approchee(epsilon):
    n = 1
    u = 1 / 2
    while 0.5 * 0.125**(n-1) > epsilon:
        n = n + 1
        u = (1+u)/(1+np.exp(u))
    return u
```

La suite commence à $n = 0$, mais l'inégalité est valable pour $n \geq 1$, donc on initialise `n` à 1 et `u` à $u_1 = \frac{1}{2}$. □

Exercice 3 : On considère la suite $(u_n)_{n\in\mathbb{N}}$ définie par : $\begin{cases} u_1 = 1 \\ \forall n \in \mathbb{N}^*,\ u_{n+1} = u_n^2 + \ln(n) + \mathrm{e}^{u_n} \end{cases}$.

Compléter la fonction **Python** suivante (dont on donne les deux premières lignes) pour qu'elle

- prenne en argument un réel `A` > 0.
- renvoie le premier entier $n \in \mathbb{N}$ tel que $u_n > A$.

```
import numpy as np
def CalcEntier(A):
```

Démonstration.

```
import numpy as np
def CalcEntier(A):
    n = 1
    u = 1
    while u <= A :
        u = u**2 + np.log(n) + np.exp(u)
        n = n + 1
    return n
```

□

Remarque 1. La suite précédente croit extrêmement vite, du fait du terme e^{u_n} dans la relation de récurrence. Python renvoie les résultats suivants :

$$\begin{aligned} u_1 &= 1 \\ u_2 &\simeq 3,7 \\ u_3 &\simeq 56,3 \\ u_4 &\geqslant 10^{24} \end{aligned}$$

et indique que $u_5 = +\infty$ (le calcul de u_5 dépasse déjà sa capacité de calcul !).