Nom	Prénom	•
140111,	ı ı ciioiii	•

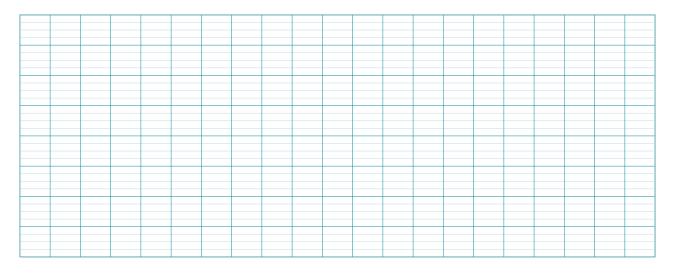
NSI2

## **Interrogation 01**

09/2021

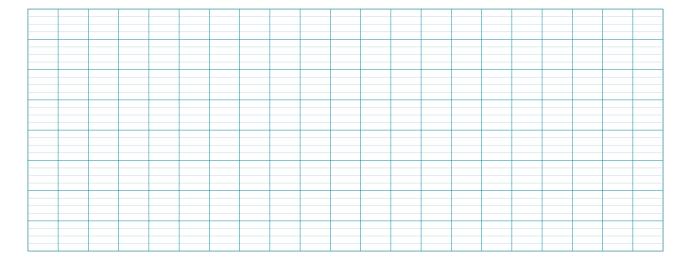


**1.** Explique *brièvement* (la réponse doit rentrer dans le cadre) et sans donner d'exemple ce qu'est une fonction récursive.



- 2. Écris ici une version РУТНОМ de la fonction récursive fibo qui
- en entrée prend un entier positif n;
- en sortie renvoie la valeur de  $F_n$  définie par

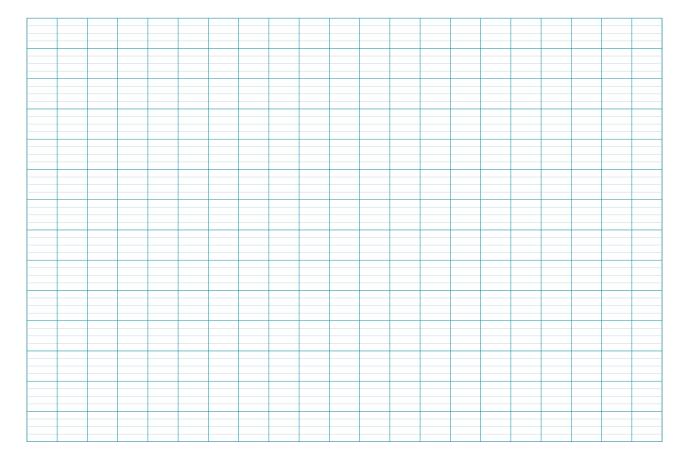
$$F_n = \begin{cases} 1 & \text{si } n = 0 \text{ ou } n = 1 \\ F_{n-1} + F_{n-2} & \text{sinon} \end{cases}$$



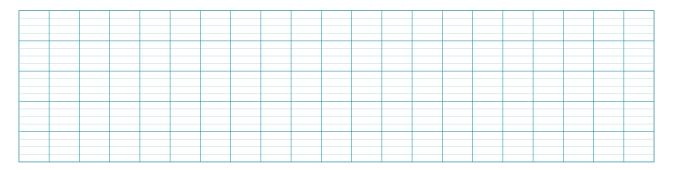
**3.** Voici une première fonction récursive :

```
def mystery1(n: int) -> int:
"""n est un entier positif"""
if int == 0:
    return 1
else:
    return 2 * mystery1(n - 1)
```

Calcule mystery1(0), mystery1(1) et mystery1(2).



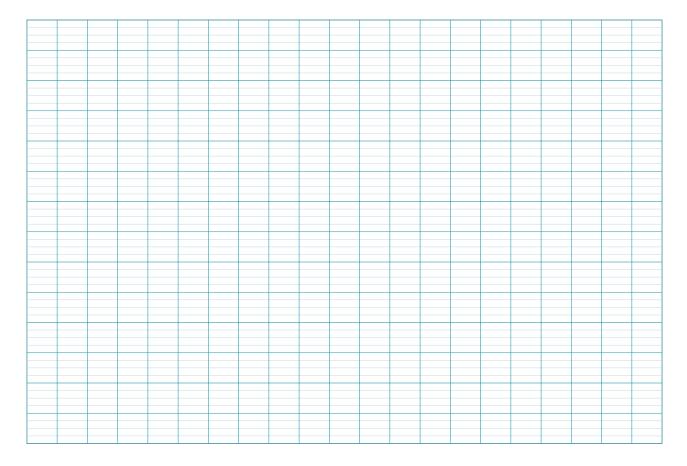
Explique ce que fait cette fonction.



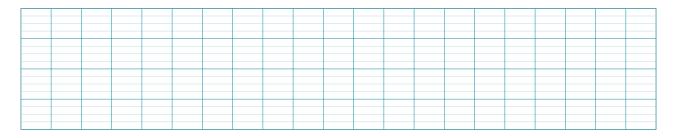
## 4. En voici une deuxième :

```
def mystery2(lst: list) -> int:
"""lst est une liste d'int "éventuellement vide"""
if lst != []:
    return lst[0] + mystery2(lst[1:])
    # on rappelle que lst[1:] désigne la liste
    # composée de lst[1],lst[2], etc jusqu'au dernier élément de lst
else:
    return 0
```

Explique ce que renvoient mystery2([]) et mystery2([7, 3, 5]).



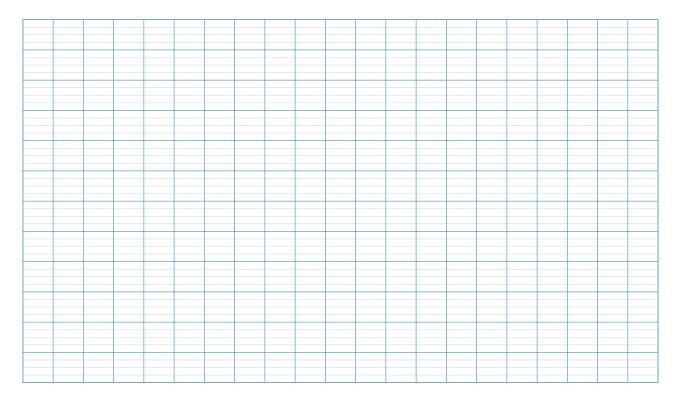
Explique ce que fait cette fonction.



## 5. En voici une dernière :

```
def mystery3(lst: list) -> bool:
"""lst est une liste d'int non vide"""
if len(lst) > 1:
    if lst[0] > lst[1]:
        return False
    else:
        return mystery3(lst[1:])
        # on rappelle que lst[1:] désigne la liste composée
        # de lst[1],lst[2], etc jusqu'au dernier élément de lst
    else:
        return True
```

Explique ce que renvoient mystery3([8]), mystery3([1,2]) et mystery3([4,1,2]).



Explique ce que fait cette fonction.

