Interrogation d'ITC n°1

Semaine du 23/09/2024

Durée : 25 minutes

Nom :		**Prénom :**	

Consignes

- Les codes doivent être présentés en mentionnant explicitement l'indentation, au moyen par exemple de barres verticales sur la gauche.
- Pour qu'un code soit compréhensible, il convient de choisir des noms de variables les plus explicites possibles.
- La performance du code proposé à chaque question entrera dans l'évaluation, de-même que l'utilisation de boucles appropriées.
- Vous pouvez bien sûr utiliser une feuille de brouillon en parallèle.

/20

Exercice 1 **Listes** [Sol 1] On considère la liste :

```
L = [6, 7.0, [1, 2, 3], "MPSI"]
```

1. /0.5 Que renvoie `L[1]` ?
2. /0.5 Que renvoie `L[-2]` ?
3. /0.5 Que renvoie `len(L)` ?
4. /0.5 Que renvoie `L[:3]` ?
5. /1 Que renvoie `L[::2]` ?
6. /1 Comment accéder à 3 stocké dans la sous-liste ?
7. /1 Que renvoie `L[1:][1][1]` ? *On expliquera soigneusement le résultat.*
8. /1 Que devient la liste après avoir exécuté la commande ci-après ?

```
L[2:2] = ["PCSI"]
```

9. /1 Dans cette question, on suppose à nouveau que `L` est la liste définie en début d'exercice. Que devient la liste après avoir exécuté la commande ci-après ?

```
L = L[:1] + ["PTSI"] + L[1:]
```

10. /1 Dans cette question, on suppose à nouveau que `L` est la liste définie en début d'exercice. Écrire une ou plusieurs commandes pour que `L` soit transformée en :

```
L = ["PCSI", 6, 7.0, "MPSI"]
```

Exercice 2 **Suite récurente d'ordre 1** [Sol 2] On considère la suite $(u_n)_n$ avec $u_0 = 2$ et :

$$\forall n \in \mathbb{N}, \quad u_{n+1} = 3u_n^2 + 4.$$

1. /3 Compléter la fonction ci-dessous afin qu'elle renvoie la valeur de u_n.

```
def suiteU(n:int)->float:
    r = ______
    for k in range(______):
        r = ______
    return r
```

2. /3 Écrire une fonction `sommeU(n)` qui renvoie la somme $u_0 + u_1 + \ldots + u_n$ sans utiliser la fonction `suiteU` écrite plus haut. *On précisera notamment la signature de la fonction.*

Exercice 3 **Fonction mystère** [Sol 3] On considère la fonction suivante :

```
def inconnue(a:float, b:float, n:int)->float:
    u, v = a, b
    if n == 0:
        return u
    elif n == 1:
        return v
    else:
        for k in range(2, n+1):
            c = u
            u = v
            v = 3*c+2*v
        return v
```

1. /1 Que renvoie l'instruction `inconnue(1, 2, 1)` ?

2. /3 Donner les différents contenus des variables `u`, `v` et `c` lors de l'exécution de `inconnue(1, 2, 4)`. *On indiquera notamment les résultats dans un tableau dont les colonnes correspondent aux variables* `u`, `v` *et* `c`. *On rappelle que l'itération* $i = 0$ *correspond à l'état des variables juste avant l'entrée dans la boucle.*

i	k_i	c_i	u_i	v_i
0	…	…	…	…

3. /2 Préciser la suite (v_n) dont les valeurs sont calculées par la fonction `inconnue`.

SOLUTIONS DES EXERCICES

Solution 1

```
>>> L = [6, 7.0, [1, 2, 3], "MPSI"]
>>> L[1] # Q1
7.0
>>> L[-2] # Q2
[1, 2, 3]
>>> len(L) # Q3
4
>>> L[:3] # Q4
[6, 7.0, [1, 2, 3]]
>>> L[::2] # Q5
[6, [1, 2, 3]]
>>> L[2][-1] # Q6
3
>>> L[1:][1][1] # Q7
2
>>> L[2:2] = ["PCSI"]
>>> L # Q8
[6, 7.0, 'PCSI', [1, 2, 3], 'MPSI']
>>> L = [6, 7.0, [1, 2, 3], "MPSI"]
>>> L = L[:1] + ["PTSI"] + L[1:]
>>> L # Q9
[6, 'PTSI', 7.0, [1, 2, 3], 'MPSI']
>>> L = [6, 7.0, [1, 2, 3], "MPSI"]
>>> del L[2]
>>> L = ["PCSI"] + L
>>> L # Q10
['PCSI', 6, 7.0, 'MPSI']
>>> L = [6, 7.0, [1, 2, 3], "MPSI"] # Autre solution :
>>> L = ["PCSI"] + L[0:2] + ["MPSI"]
>>> L
['PCSI', 6, 7.0, 'MPSI']
```

Solution 2

1.
```
def suiteU(n:int)->float:
    r = 2
    for _ in range(1, n+1):
        r = 3*(r**2)+4
    return r
```

2.
```
def sommeU(n:int)->float:
    r = 2
    S = r
    for _ in range(1, n+1):
        r = 3*(r**2)+4
        S += r
    return S
```

Solution 3

1. Que renvoie l'instruction `inconnue(1, 2, 1)` ? Ici `n = 1`, donc on rentre dans le deuxième test et la fonction renvoie `v` donc `b`, donc 2.

2.

i	k_i	c_i	u_i	v_i
0			1	2
1	2	1	2	7
2	3	2	7	20
3	4	7	20	61

3. Les trois étapes de la boucles peuvent être comprises comme :

`u, v = v, 3*u+2*v`.

Le programme code la suite ci-dessous :

$$\begin{cases} v_0 = a,\ v_1 = b, \\ \forall n \in \mathbb{N},\ v_{n+2} = 3v_n + 2v_{n+1}. \end{cases}$$

On peut montrer par récurrence double la formule ci-dessous :

$$\forall n \in \mathbb{N}, \quad v_n = \frac{1}{4}\left((-1)^n + 3^{n+1}\right).$$