TP 02 – Calcul des termes d'une suite

I Suite définie de manière explicite

Exercice 1 On considère la suite $(u_k)_{k\in\mathbb{N}}$ définie par

$$\forall k \in \mathbb{N}, \qquad u_k = k^2 + 1$$

On donne également le programme Python suivant

Entrée [1]:
```python
for k in range(0, 5):
    u = k**2+1
```

1. Compléter le tableau suivant.

Valeurs de k successives	Valeurs de u successives	Terme de la suite $(u_n)_{n\in\mathbb{N}}$ correspondant
0	1	u_0
1	2	u_1
2	5	u_2
3	10	u_3
4	17	u_4

2. Comment modifier le programme de départ pour qu'il affiche **tous** les termes de la suite $(u_n)_{n\in\mathbb{N}}$ pour n allant de 0 à 4 ?

Entrée [2]:
```python
for k in range(0, 5):
    u = k**2+1
    print(u)
```

Out [2]:
```
1
2
5
10
17
```

3. Comment modifier le programme de départ pour qu'il affiche **tous** les termes de la suite $(u_n)_{n\in\mathbb{N}}$ pour n allant de 0 à 4 au sein d'une **liste** ?

Entrée [3]:
```python
L = []
for k in range(0, 5):
    u = k**2+1
    L.append(u)
print(L)
```

Out [3]:
```
[1, 2, 5, 10, 17]
```

4. Comment modifier le programme de départ pour qu'il affiche **tous** les termes de la suite $(u_n)_{n\in\mathbb{N}}$ pour n allant de 0 à 11 au sein d'une **liste** ?

Entrée [4]:
```
L = []
for k in range(0, 12):
    u = k**2+1
    L.append(u)
print(L)
```

Out [4]:
```
[1, 2, 5, 10, 17, 26, 37, 50, 65, 82, 101, 122]
```

5. Comment modifier le programme de départ pour qu'il affiche **tous** les termes de la suite $(u_n)_{n\in\mathbb{N}}$ pour n allant de 8 à 11 au sein d'une **liste** ?

Entrée [5]:
```
L = []
for k in range(8, 12):
    u = k**2+1
    L.append(u)
print(L)
```

Out [5]:
```
[65, 82, 101, 122]
```

6. Écrire une fonction, appelée `listesuite1`, qui prend en argument un entier `n` et qui renvoie la liste de tous les termes de la suite (depuis u_0) jusqu'au `n`-ième. *On vérifiera que l'évaluation de* `listesuite1` *en* `11` *donne une liste dont le premier terme est* `1` *et le dernier terme est* `122`.

Entrée [6]:
```
def listesuite1(n):
    L=[]
    for k in range(0, n+1):
        u = k**2+1
        L.append(u)
    return(L)
```

Entrée [7]:
```
listesuite1(11)
```

Out [7]:
```
[1, 2, 5, 10, 17, 26, 37, 50, 65, 82, 101, 122]
```

II Suite définie de manière récurrente

Exercice 2 On considère la suite $(u_k)_{k\in\mathbb{N}}$ définie par

$$u_0 = 0 \quad \text{et} \quad \forall k \in \mathbb{N}, \qquad u_{k+1} = (u_k)^2 + 1$$

On donne également le programme Python suivant

Entrée [8]:
```
u = 0
for k in range(1, 5):
    u = u**2 + 1
```

1. Compléter le tableau suivant et l'affichage du programme.

Valeurs de k successives	Valeurs de u successives	Terme de la suite $(u_n)_{n\in\mathbb{N}}$ correspondant
———	0	u_0
1	1	u_1
2	2	u_2
3	5	u_3
4	26	u_4

2. Comment modifier le programme de départ pour qu'il affiche **tous** les termes de la suite $(u_n)_{n\in\mathbb{N}}$ pour n allant de 0 à 4 ?

Entrée [9]:
```
u = 0
print(u)
for k in range(1, 5):
    u = u**2 + 1
    print(u)
```

Out [9]:
```
0
1
2
5
26
```

3. Comment modifier le programme de départ pour qu'il affiche **tous** les termes de la suite $(u_n)_{n\in\mathbb{N}}$ pour n allant de 0 à 4 au sein d'une **liste** ?

Entrée [10]:
```
L=[]
u = 0
L.append(u)
for k in range(1, 5):
    u = u**2+1
    L.append(u)
print(L)
```

Out [10]:
```
[0, 1, 2, 5, 26]
```

4. Comment modifier le programme de départ pour qu'il affiche **uniquement** le terme u_4 de la suite $(u_n)_{n\in\mathbb{N}}$?

Entrée [11]:
```
u = 0
for k in range(1, 5):
    u = u**2 + 1
print(u)
```

Out [11]:
```
26
```

5. Écrire une fonction, appelée `suite2`, qui prend en argument un entier `n` et qui renvoie le terme u_n de la suite. *On vérifiera que l'évaluation de* `suite2` *en* `4` *donne* `26`.

Entrée [12]:
```
def suite2(n):
    u = 0
    for k in range(1, n+1):
        u = u**2+1
    return(u)
```

Entrée [13]:
```
suite2(4)
```

Out [13]:
```
26
```

6. Écrire une fonction, appelée `listesuite2`, qui prend en argument un entier `n` et qui renvoie la liste de tous les termes de la suite (depuis u_0) jusqu'au `n`-ième. *On vérifiera que l'évaluation de* `listesuite1` *en* `4` *donne* `[0, 1, 2, 5, 26]`.

Entrée [14]:
```
def listesuite2(n):
    u=0
    L=[0]
    for k in range(1, n+1):
        u = u**2+1
        L.append(u)
    return(L)
```

Entrée [15]:
```
listesuite2(4)
```

Out [15]:
```
[0, 1, 2, 5, 26]
```

III Exercices supplémentaires

Exercice 3 On considère la suite $(u_k)_{k\in\mathbb{N}}$ définie par

$$\forall k \in \mathbb{N}, \qquad u_k = \frac{1}{k^2+1}$$

Écrire une fonction, appelée `exo1`, qui prend en argument un entier `n` et qui renvoie la liste de tous les termes de la suite (depuis u_0) jusqu'au `n`-ième. *On vérifiera que l'évaluation de* `exo1` *en* `3` *donne une liste finissant par* `0.1`.

Entrée [16]:
```
def exo1(n):
    L=[]
    for k in range(0, n+1):
        u = 1/(k**2+1)
        L.append(u)
    return(L)
```

Entrée [17]:
```
exo1(3)
```

Out [17]:
```
[1.0, 0.5, 0.2, 0.1]
```

Exercice 4 On considère la suite $(u_k)_{k\in\mathbb{N}}$ définie par

$$u_0 = 0 \quad \text{et} \quad \forall k \in \mathbb{N}, \qquad u_{k+1} = \sqrt{u_k + 2}$$

1. Écrire une fonction, appelée `exo2`, qui prend en argument un entier `n` et qui renvoie le terme u_n de la suite. *On vérifiera que l'évaluation de* `exo2` *en* `4` *donne environ* `1.99`.

Entrée [18]:
```
import numpy as np

def exo2(n):
    u = 0
    for k in range(1, n+1):
        u = np.sqrt(u+2)
    return(u)
```

Entrée [19]:
```
exo2(4)
```

Out [19]:
```
1.9903694533443939
```

2. Écrire une fonction, appelée `listeexo2`, qui prend en argument un entier `n` et qui renvoie la liste de tous les termes de la suite (depuis u_0) jusqu'au `n`-ième. *On vérifiera que l'évaluation de* `listeexo2` *en* `4` *donne une liste finissant par environ* `1.96`.

Entrée [20]:
```
import numpy as np

def listeexo2(n):
    u=0
    L=[0]
    for k in range(1, n+1):
        u = np.sqrt(u+2)
        L.append(u)
    return(L)
```

Entrée [21]:
```
listeexo(3)
```

Out [21]:
```
[0, 1.4142135623730951, 1.8477590650225735, 1.9615705608064609]
```

Exercice 5 On considère la suite $(S_n)_{n\in\mathbb{N}}$ définie par

$$\forall n \in \mathbb{N}, \qquad S_n = \sum_{k=0}^{n} k$$

1. Donner, pour tout $k \in \mathbb{N}$, une relation entre S_{k+1} et S_k.

   On a,
   $$\forall k \in \mathbb{N}, \qquad S_{k+1} = S_k + (k+1)$$

2. Écrire une fonction, appelée `sommeentiers`, qui prend en argument un entier `n` et qui renvoie la valeur de la somme S_n. *On vérifiera que l'évaluation de* `sommeentiers` *en* `6` *donne* `21`.

Entrée [22]:
```
def sommeentiers(n):
    S=0
    for k in range(0,n):
        S = S + k+1
    return(S)
```

Entrée [23]:
```
sommeentiers(6)
```

Out [23]:
```
21
```

Exercice 6 Écrire un programme calculant $S = \sum_{k=1}^{1000} \frac{1}{k^2}$. Vérifier que $S \approx 1.64$.

Entrée [24]:
```
S=0
for k in range(0,1000):
    S = S + 1/(k+1)**2
print(S)
```

Out [24]:
```
1.6439345666815615
```

Exercice 7 Trouver, pour tout $k \in \mathbb{N}$, une relation entre $(k+1)!$ et $k!$. En déduire le script d'une fonction, appelée `factorielle`, qui prend en argument un entier `n` et qui renvoie la valeur de `n!`. *On vérifiera que* `factorielle(5)` *renvoie* `120`.

Entrée [25]:
```
def factorielle(n):
    P=1
    for k in range(1,n+1):
        P=P*k
    return(P)
```

Entrée [26]:
```
factorielle(5)
```

Out [26]:
```
120
```