Python-ECG1-13-revisions-cor

December 15, 2021

1 TP13 - Révisions - corrigé

1.1 Exercice 1

On considère la suite $(u_n)_{n\in\mathbb{N}}$ telle que $u_0 = 1$ et

$$\forall n \in \mathbb{N}, \quad u_{n+1} = 2(\ln(2))^{n+1} - (n+1)u_n.$$

1) Écrire une fonction qui, étant donné n, renvoie la valeur de u_n.

[1]:
```
import numpy as np
def SuiteU(n):
    u = 1
    for k in range(n):
        u = 2*np.log(2)**(k+1) - (k+1)*u
    return u
```

2) Écrire des instructions qui permettent de créer la liste $L = [u_0, u_1, \ldots, u_{20}]$. La suite (u_n) semble t-elle convergente ?

[2]:
```
L = [ SuiteU(n) for n in range(21)]
print(L)

# ou
u = 1
L = [u]
for k in range(20):
    u = 2*np.log(2)**(k+1) - (k+1)*u
    L.append(u)
print(L)
```

```
[1, 0.3862943611198906, 0.18831730559662163, 0.10109738718799399,
0.05728064841419089, 0.03360215344332812, 0.020197917004726423,
0.012362137711838195, 0.007672583781018122, 0.004815222991494922,
0.003049496664176997, 0.0019458691121755842, 0.0012495945066498607,
0.0008067085887976388, 0.0005252353593497381, 0.0003388399223764886,
0.0002564050553949539, -0.007222815760383094, 0.13273895963557145,
-2.5201491362942434, 50.40429353428721]
[1, 0.3862943611198906, 0.18831730559662163, 0.10109738718799399,
0.05728064841419089, 0.03360215344332812, 0.020197917004726423,
```

```
0.012362137711838195, 0.007672583781018122, 0.004815222991494922,
0.003049496664176997, 0.0019458691121755842, 0.0012495945066498607,
0.0008067085887976388, 0.0005252353593497381, 0.00031388399223764886,
0.0006564050553949539, -0.007222815760383094, 0.13273895963557145,
-2.5201491362942434, 50.40429353428721]
```

3) Déterminer le premier entier n tel que $u_n < -10^4$.

[7]:
```
# Solution 1
n = 0
while SuiteU(n) >= -10**(4):
    n = n+1
print(n)

# Solution 2
n = 0
u = 1
while u >= -10**(4):
    u = 2*np.log(2)**(n+1) - (n+1)*u
    n = n+1
print(n)
```

```
23
23
```

4) Déterminer le premier entier n tel que $u_n > 10^4$.

[5]:
```
# Solution 1
n = 0
while SuiteU(n) <= 10**(4):
    n = n+1
print(n)

# Solution 2
n = 0
u = 1
while u <= 10**(4):
    u = 2*np.log(2)**(n+1) - (n+1)*u
    n = n+1
print(n)
```

```
22
22
```

1.2 Exercice 2

1) Créer une liste A de 20 entiers aléatoires entre 1 et 100.

[12]:
```
import numpy.random as rd
A = [rd.randint(1,101) for k in range(20)]
print(A)
```

```
[10, 84, 39, 24, 18, 95, 19, 10, 34, 31, 20, 42, 83, 72, 27, 75, 41, 26, 47, 53]
```

2) Déterminer la somme des éléments de A.

[13]:
```
# Version 1
S = 0
for k in range(len(A)):
    S = S + A[k]
print(S)

# Version 2
S = 0
for x in A:
    S = S + x
print(S)
```

```
850
850
```

3) Déterminer la valeur maximale de A.

[14]:
```
# Version 1
M = A[0]
for k in range(1, len(A)):
    if A[k] > M:
        M = A[k]
print(M)

# Version 2
M = A[0]
for x in A:
    if x > M:
        M = x
print(M)
```

```
95
95
```

1.3 Exercice 3

On souhaite simuler l'expérience aléatoire suivante. On considère $n \geqslant 2$ personnes (numérotées de 0 à $n-1$). Ces personnes amènent toute un cadeau (numéroté de la même façon). Ces cadeaux sont mis en commun puis chaque personne, dans l'ordre de la numérotation, tire au hasard un cadeau. Si elle tombe sur le sien, elle en choisit un autre, saud la dernière personne qui prend le cadeau restant.

Écrire une fonction **SecretSanta(n)** qui renvoie une liste S formée des cadeaux piochés, dans l'ordre.

Indications : on utilisera la bibliothèque **numpy.random**

rd.choice(liste) : choisit un élément au hasard dans une liste donnée

liste.remove(x) : enlève l'élément x d'une liste donnée

La liste des cadeaux initiale pourra être créée avec **cadeaux = [i for i in range(n)]**

[10]:
```
import numpy.random as rd

def SecretSanta(n):
    S = []
    cadeaux = [i for i in range(n)]
    #print(cadeaux)
    for k in range(n-1):
        # La personne k choisit un cadeau
        c = rd.choice(cadeaux)
        while c == k : # si la personne k tire son cadeau
            c = rd.choice(cadeaux) # elle en prend un autre.
        # on répète ceci autant de fois que nécessaire, d'où le while
        S.append(c)
        cadeaux.remove(c)
    # dernière personne
    S = S + cadeaux
    return S

print(SecretSanta(10))
```

```
[7, 5, 9, 6, 0, 4, 2, 3, 1, 8]
```

On supose que $n = 4$. Faire 1000 simulations et compter le nombre de fois où la dernière personne a son cadeau. En déduire une valeur approchée de la probabilité que la dernière personne tombe sur son cadeau.

[16]:
```
n = 4
compteur = 0
for k in range(1000):
    Simul = SecretSanta(n)
    if Simul[-1] == n-1:
        compteur = compteur + 1
print(compteur)

print("proba =", compteur/1000)
```

```
123
proba = 0.123
```

[8]:
```
5/36
```

[8]: 0.1388888888888889