Concours blanc - TP d'informatique - Sujet 1 - Correction

Mardi 30 mai 2023

1. ***1 points***

```
def presence(L,x):
    for e in L:
        if x==e:
            return True
    return False
```

2. (a) ***1,5 points***

```
def suite(n):
    S = 0
    for k in range(1,n+1):
        S += (1/k**2)
    return S
```

(b) ***4 points***

```
import matplotlib.pyplot as plt

def graphique(n):
    X = [k for k in range(1,n+1)]
    Y = []
    S = 0
    for k in range(1,n+1):
        S += (1/k**2)
        Y.append(S)
    plt.plot(X,Y,'*')
    plt.show()
```

(c) ***1,5 points***

On fait afficher graphique(100) .

Par lecture graphique, il semble que (u_n) ait une limite. Celle-ci serait un réel légèrement supérieur à 1,6.

3. (a) ***2,5 points***

```
def cumul(N):
    res = []
    S = 0
    n = len(N)
    for k in range(0,n):
        S+= N[k]
        res.append(S)
    return res
```

(b) ***1 points***

```
def maximum(L):
    M = L[0]
    for e in L:
        if e>M:
            M = e
    return M
```

(c) ***3 points***

```
def mode(X,N):
    M = N[0]
    indice_M = 0
    n = len(N)
    for k in range(0,n):
        if N[k]>M:
            M = N[k]
            indice_M = k
    return X[k]
```

4. (a) ***1,5 points***

On pose $f : t \mapsto (1-t)^5 e^{-2t}$.

Par les théorèmes opératoires, f est continue sur $[0,1]$.

Donc I est bien définie.

(b) ***1,5 points***

$$I = \lim_{n\to+\infty} \frac{1}{n-1} \sum_{k=0}^{n} \left(1-\frac{k}{n}\right)^5 e^{-2k/n}$$

(c) ***2,5 points***

```
1 from math import *
2
3 def estimation(n):
4     S = 0
5     for k in range(n):
6         S += ((1-(k/n))**5)*exp(-2*(k/n))
7     return S/n
```

Concours blanc - TP d'informatique - Sujet 2 - Correction

Mardi 30 mai 2023

1. ***1 points***

```
def comptage(L,x):
    c = 0
    for e in L:
        if e==x :
            c += 1
    return c
```

2. (a) ***2,5 points***

```
from math import *

def suite(n):
    if n==0:
        return 0
    else:
        return sqrt(2-suite(n-1))
```

   (b) ***1,5 points***

   Le message d'erreur est "maximum recursion depth exceeded in comparison".
   Cela veut dire qu'on ne peut pas faire autant d'appels récursifs à une même fonction.

3. (a) ***1 points***

```
def maximum(L):
    M = L[0]
    for e in L:
        if e>M:
            M=e
    return M
```

   (b) ***4 points***

```
def occurrences(L):
    M = L[0]
    compteur = 0
    for e in L:
        if e==M :
            compteur +=1
        elif e>M:
            M = e
            compteur = 1
    return compteur
```

4. (a) ***4 points***

   On définit la fonction f par :

$$\begin{array}{rccc} f & : & [0,1] & \to & \mathbb{R} \\ & & x & \mapsto & x^5+2x^2+x-1 \end{array}$$

   Par les théorèmes opératoires, f est continue sur $[0,1]$.
   Par les théorèmes opératoires, f est dérivable sur $[0,1]$.
   Pour tout $x \in [0,1]$, $f'(x) = 5x^4+4x+1 > 0$.
   Donc f est continue strictement croissante sur $[0,1]$.
   Le théorème de la bijection s'applique.
   Donc f est bijective de $[0,1]$ dans $f\Big([0,1]\Big) = [\,f(0,f(1)\,] = [-1,3]$.
   En particulier, 0 a un unique antécédent par f dans $[0,1]$.
   C'est-à-dire : (E) a une unique solution dans I.

(b) ***3 points***

```python
import numpy as np
import matplotlib.pyplot as plt

def graphique():
    X = np.linspace(0,1,100)
    Y = [x**5+2*x**2+x-1 for x in X]
    plt.plot(X,Y)
    plt.grid()
    plt.show()
```

(c) ***3 points***

```python
def f(x):
    return x**5+2*x**2+x-1

def dichotomie(e):
    a = 0
    b = 1
    while (b-a) >= e:
        c = (a+b)/2
        if f(a)*f(c) < 0:
            b = c
        else:
            a = c
    return (a,b)
```

Concours blanc - TP d'informatique - Sujet 3 - Correction

Mardi 30 mai 2023

1. ***2 points***

```
def suite_liste(n):
    return [1/(k**2+1) for k in range(0,n+1)]
```

2. (a) ***2,5 points***

```
def etendue(X):
    m = X[0]
    M = X[0]
    for e in X:
        if e>M:
            M=e
        if e<m:
            m=e
    return M-m
```

(b) ***2 points***

```
def moyenne(X):
    S = 0
    for e in X:
        S += e
    return S/len(X)
```

(c) ***2 points***

La variance de x est :

$$s_x^2 = \overline{x^2} - (\overline{x})^2$$

$$= \frac{1}{n}\sum_{k=1}^{n} x_k^2 - \left(\frac{1}{n}\sum_{k=1}^{n} x_k\right)^2$$

(d) ***3,5 points***

```
def variance(X):
    somme_x = 0
    somme_carre = 0
    n = len(X)
    for k in range(0,n):
        somme_x += X[k]
        somme_carre += X[k]**2
    return (1/n)*somme_carre - ((1/n)*somme_x)**2
```

3. ***2,5 points***

```
import matplotlib.pyplot as plt
import numpy as np

def saturation_R(im):
    n = im.shape[0]
    p = im.shape[1]
    for i in range(n):
        for j in range(p):
            G = im[i,j][1]
            B = im[i,j][2]
            im[i,j] = [255,G,B]
    return im
```

4. (a) ***1 points***

n \ k	0	1	2	3	4
0	1	0	0	0	0
1	1	1	0	0	0
2	1	2	1	0	0
3	1	3	3	1	0
4	1	4	6	4	0

(b) ***1,5 points***

Soient $n \in \mathbb{N}$ et $k \in [\![0, n]\!]$.

$\binom{n}{k} + \binom{n}{k+1} = \binom{n+1}{k+1}$

(c) ***3 points***

```
1 def coeff_bin(n,k):
2     if k>n:
3         return 0
4     if k==0:
5         return 1
6     else:
7         return coeff_bin(n-1,k-1)+coeff_bin(n-1,k)
```