

I. Analyse : extraits de divers sujets de concours Edhec, EML...

1. Un grand classique

On considère la suite $(u_n)_{n \geq 0}$ définie par :

$$u_0 = 0, \quad u_1 = 1 \quad \text{et} \quad \forall n \in \mathbb{N}, \quad u_{n+2} = u_{n+1} + u_n,$$

Comment s'appelle cette suite ???

Recopier et compléter la fonction Python suivante afin que, prenant en argument un entier $n \geq 2$, elle calcule et renvoie la valeur du terme u_n de la suite $(u_n)_{n \geq 0}$.

```
def suite(n):
    v=0
    w=1
    for k in range(2,n+1):
        .....
        .....
        .....
    end
    return .....
```

2. Soit $n \in \mathbb{N}$. On définit une fonction polynômiale $P_n : \mathbb{R} \rightarrow \mathbb{R}$, par :

$$\forall x \in \mathbb{R}, \quad P_n(x) = \sum_{k=0}^{2n+1} \frac{(-x)^k}{k!}$$

En Python, après installation du package `math` : `import math as math`, l'instruction `math.factorial(n)` permet de calculer $n!$.

Ecrire une fonction Python d'en-tête `def P(n,x) :` qui prend pour arguments un entier naturel n et un réel x et qui renvoie la valeur de $P_n(x)$.

On admet que la fonction P_n est strictement décroissante sur $\mathbb{R}$, et qu'elle ne s'annule qu'une seule fois sur l'intervalle $[1, 2n+1]$.

Recopier et compléter la fonction Python suivante afin que, prenant pour argument un entier n de $\mathbb{N}$, elle renvoie une valeur approchée de u_n à 10^{-3} près à l'aide de la **méthode par dichotomie**.

```
1. def suite(n):
2.     a = .....
3.     b = .....
4.     c = (a+b)/2
5.     while .....
6.         if ..... :
7.             a = c
8.         else :
9.             b = c
10.    c = .....
12.    return .....
```

3. On considère une suite $(u_n)_{n \in \mathbb{N}}$ définie par $u_0 = 1$ et pour tout $n \in \mathbb{N}$,

$$u_{n+1} = \sqrt{u_n^2 + u_n}$$

Pour tout entier n , on pose

$$S_n = \sum_{k=0}^n u_k$$

On admet que $\lim_{n \rightarrow +\infty} S_n = +\infty$.

Ecrire un programme qui affiche la plus petite valeur de n pour laquelle $S_n > 1000$.

4. On considère la suite (u_n) définie par : pour tout $n \geq 1$, $u_n = \prod_{k=1}^n (1 - \frac{1}{2k})$

(a) Ecrire une fonction `def u(n) :` qui calcule le n -ème terme de la suite (u_n) .

(b) On admet que la suite (u_n) tend vers 0. Ecrire un programme qui affiche le premier entier n tel que $u_n \leq 10^{-3}$.

II. Calcul matriciel

1. Soit

$$B = \begin{pmatrix} 2 & -2 & -5 \\ -2 & 5 & 10 \\ 1 & -2 & -4 \end{pmatrix}$$

On effectue des calculs avec Python, les résultats suivants s'affichent

```
>>> B=np.array([[2,-2,-5],[-2,5,10],[1,-2,-4]])
>>> import numpy.linalg as al
>>> al.matrix_rank(B)
3.
>>> al.matrix_power(B-np.eye(3),3)
array([[0., 0., 0.],
       [0., 0., 0.],
       [0., 0., 0.]])
>>> al.matrix_rank(B-np.eye(3))
1
```

Que peut-on en déduire ?

Justifier que la matrice B n'est pas diagonalisable.

2. Ecrire une fonction Python d'intitulé `def A(n) :` qui renvoie la matrice

$$A = \begin{pmatrix} 1 & 1 & 0 & 0 & \dots & \dots & \dots & 0 \\ 0 & 2 & 1 & 0 & \ddots & & & 0 \\ \vdots & \ddots & 3 & 1 & \ddots & \ddots & & \vdots \\ \vdots & & \ddots & 4 & \ddots & 0 & \ddots & \vdots \\ \vdots & & & \ddots & \ddots & 1 & \ddots & 0 \\ 0 & & & (0) & \ddots & k & \ddots & 0 \\ 0 & & & & \dots & \dots & \ddots & 1 \\ 0 & \dots & \dots & \dots & \dots & \dots & 0 & n \end{pmatrix} \in \mathcal{M}_n(\mathbb{R})$$