Travaux Pratiques 5 : Tableaux (= Vecteurs = Matrices lignes)

Commencer par importer la bibliothèque `numpy` : `import numpy as np`.

Exercice 1: *Prise en main*

A l'aide du cours, deviner l'action des commandes suivantes. Si besoin, les taper ensuite dans la console **dans l'ordre** pour vérifier et annoter vos feuilles !

```
x=np.arange(1,10)
x # dans la console, print est inutile
y=np.arange(1,10,2)
y
z=np.linspace(1,10,3)
z
t=np.linspace(0,1,10)
t
a=np.array([1,3,8])
a
np.shape(a)
a[1]
a[1:]
np.min(a)
np.max(a)
np.sum(a)
np.prod(a)
```

Exercice 2:

Opérations coefficients par coefficients sur les tableaux. Comme pour l'exercice 1, deviner les résultats avant de taper les instructions dans la console !

```
x=np.array([-1,1,5])
2+x
3*x
np.exp(x)
x*x
x**2
2.**x # point nécessaire pour que 2 soit
      # vu comme un réel et non un entier
1/x
y=np.array([0,-1,-2])
x*y
y/x
```

Exercice 3:

```
C=np.ones(5)
C[2]=0
C
```

Exercice 4: *A vous de jouer*

Créer les vecteurs suivants, sans rentrer les coefficients à la main :

a=[1,2,...,99,100], b=[2,4,6,...,48,50], c=[20,19,18,...,1], d=[1.5,2.5,...,10.5],
e= vecteur contenant 50 coefficients allant de 1 à 30 équidistants, f=(5,5,...,5) de taille 50

Exercice 5:

1. Créer le vecteur `x=[8,13,11,7,5,4,9,1]`.
2. Ecrire une commande pour récupérer l'élément 8, puis une autre pour récupérer l'élément 7.
3. Ecrire une commande pour remplacer 4 par 3.
4. Ecrire une commande pour créer le vecteur $[e^8, e^{13}, e^{11}, e^7, e^5, e^3, e^9, e]$.

Exercice 6: <u>*Important !*</u>

A l'aide de la commande `k=np.arange(début,fin +1,pas)` et SANS boucle for, créer :

1. le tableau de taille 20 qui comporte les 20 premiers entiers non-nuls au carré.
2. (a) une fonction qui prend en entrée un entier naturel n et qui renvoie la valeur de $\sum_{k=1}^{n} k^3$.
   (b) Révisions : comment avait-on fait sans les tableaux, à l'aide de `for` ? Réécrire la fonction correspondante.
3. (a) une fonction qui prend en entrée un entier naturel n et qui renvoie la valeur de $n!$.
   (b) Révisions : comment avait-on fait sans les tableaux à l'aide de `for` ? Réécrire la fonction correspondante.
4. le vecteur de taille 10 : $((-1)^1, (-1)^2, ..., (-1)^{10})$.
5. le vecteur des nombres de la forme $(-1)^n n^2$, pour $n = 1, .., 20$
6. *bonus :* une fonction qui prend en entrée deux entiers naturels non nul n et p et qui renvoie le calcul de $\sum_{k=1}^{n} k^p$.

Exercice 7: `Edhec E 2019`

Soit u la suite définie pour tout $n \in \mathbb{N}^*$ par $u_n = \frac{4^n (n!)^2}{(2n+1)!}$.

On admet que, si `t` est un vecteur, la commande `np.prod(t)` renvoie le produit des éléments de `t`. Compléter le script suivant afin qu'il permette de calculer et d'afficher la valeur de u_n pour une valeur n entrée par l'utilisateur.

```
n=int(input('entrez une valeur pour n : '))
x=np.arange(1,n+1)
m=2*n+1
y=np.arange(1,m+1)
v=------
w=------
u=------*v**2/w
print(u)
```

Exercice 8: *Suites et Vecteurs*

Soit la suite u définie par la relation de récurrence $u_0 = 1$ et $\forall n \in \mathbb{N}^*$, $u_{n+1} = e^{1/u_n}$.

1. Ecrire une fonction python qui à un entier n associe le réel u_n. (pour vérifier $u_5 \simeq 1.83$)
2. Compléter la fonction précédente afin qu'elle donne en sortie le vecteur $U = (u_0, u_1, u_2, ..., u_n)$.
   Introduire le vecteur U via `U=np.zeros(n+1)`, *puis modifier les coefficients de U via une boucle for, cf cours* .

Exercice 9:

Reprendre les questions de l'exercice précédent avec la suite u définie par : $u_0 = 2$ et $\forall n \in \mathbb{N}$, $u_{n+1} = \ln(1 + 2u_n)$. Que peut-on conjecturer sur la convergence de la suite ?

Exercice 10: `Edhec S 2017` (hors question préliminaire)

Question préliminaire :

Deviner les affichages des instructions suivantes : `2**x` et `x**2` avec `x=np.array([1,2,3])`

Soit n un entier supérieur ou égal à 1, et on considère la fonction f_n définie par : $\forall x \in [0,1]$, $f_n(x) = \sum_{k=1}^{n} x^k$.

1. (a) Compléter la fonction `python` suivante pour qu'elle renvoie la valeur de $f_n(x)$ à l'appel de `f(x,n)`, où x et n sont donnés par l'utilisateur.
   ```
   def f(x,n):
       return np.sum(------)
   ```
   (b) Transformer pour $x \neq 1$, l'expression de $f_n(x)$ puis en déduire une deuxième façon de déclarer `f`, en complétant la déclaration suivante où la fonction est toujours nommée `f` :
   ```
   def f(x,n):
       return ------ if x==1 else ------
   ```
2. Montrer que l'équation $f_n(x) = 1$ d'inconnue $x \in [0,1]$ possède une unique solution a_n dans $[0,1]$.
3. (a) Montrer que $f_{n+1}(a_n) \geq 1$ et en déduire que la suite $(a_n)_{n\in\mathbb{N}^*}$ est décroissante.
   (b) En déduire que la suite $(a_n)_{n\in\mathbb{N}^*}$ converge.
4. (a) Déterminer a_2 puis vérifier que $0 \leq a_2 < 1$.
   (b) Utiliser les variations de la suite $(a_n)_{n\in\mathbb{N}^*}$ pour établir que $\lim\limits_{n\to+\infty} a_n^{n+1} = 0$.
   (c) En déduire que $\lim\limits_{n\to+\infty} a_n = \frac{1}{2}$.
5. On suppose que f_n a été déclarée (voir question 1) et on considère les commandes supplémentaires suivantes :
   ```
   n=int(input('entrer la valeur de n :'))
   x=0
   while f(x,n)<1:
       x=x+0.001
   print(x)
   ```
   Quel est le lien entre le résultat affiché et a_n ?

Exercice 11: ** `inspiré d'Edhec S 2017` *pour les plus rapides uniquement*

Aide python : si `u` est un vecteur, la commande `np.cumprod(u)` renvoie un vecteur de même format que `u` dont le k^{ie} élément est le produit des k premiers éléments de `u`. (à tester dans la console !)

Compléter la fonction suivante afin qu'elle retourne la variable `s` contenant la valeur $k!\sum_{i=1}^{k} \frac{x^i}{i!}e^{-x}$.

```
def edhec2017(k,x):
    p=np.prod(np.arange(1,k+1))
    u=------ / ------
    s=p*------*np.exp(-x)
    return s
```

Exercice 12: *Vecteurs et probabilités*

Rappel : la commande `rd.randint(1,7)` simule un lancer de dé équilibré.

1. On suppose les bibliothèques `numpy` et `numpy.random` importées. Lire (sans exécuter) le programme suivant :
   ```
   n=int(input('entrer un entier n '))
   T=np.zeros(6)
   for k in range(n):
       i=rd.randint(1,7)
       T[i-1]=T[i-1]+1
   print(T)
   ```
2. Prenons $n = 5$ et supposons que les 5 simulations successives de lancers de dé ont donné les résultats : 2,1,6,1,1. Que contient le tableau T au départ ? Comment évolue-t-il ? Que contient-il à la fin ?
3. Retour au cas général : que contient le tableau T à la fin ?
4. Dans le cas où n est grand, donner un équivalent des valeurs contenues dans le tableau T.