

Semaine n°26 du 20 au 24 mai

Informatique(Python) : cf exemples en annexe

- ⇒ Tableau 2D, bibliothèque Numpy
- ⇒ Simulation expérience aléatoire. Estimation des probabilités et de l'espérance.

Limites de fonctions

- ⇒ Définitions avec les quantificateurs des limites suivantes :
 $\lim_{x \rightarrow x_0} f(x) = \ell$, $\lim_{x \rightarrow x_0} f(x) = +\infty$, $\lim_{x \rightarrow x_0} f(x) = -\infty$, $\lim_{x \rightarrow +\infty} f(x) = \ell$, $\lim_{x \rightarrow +\infty} f(x) = +\infty$, ...
- ⇒ Limite à gauche et limite à droite en un point, définition avec les quantificateurs.
- ⇒ Asymptote verticale et horizontale.
- ⇒ Opérations sur les limites : addition, multiplication, quotient, composition, formes indéterminées.
- ⇒ Limites et inégalités : Si $\lim_{x \rightarrow x_0} f(x) = l > 0$ alors il existe un voisinage de x_0 sur lequel f est strictement positive, théorème des gendarmes, théorèmes de comparaison.
- ⇒ Une fonction monotone sur un intervalle ouvert admet une limite en chacune des bornes de l'intervalle.
- ⇒ Croissances comparées (logarithme, puissance, exponentielle)
- ⇒ fonctions équivalentes : définition, équivalents usuels :
 - polynômes en $+\infty$, $-\infty$ et 0.
 - $\sin(x) \underset{x \rightarrow 0}{\sim} x$
 - $\tan(x) \underset{x \rightarrow 0}{\sim} x$
 - $\ln(1+x) \underset{x \rightarrow 0}{\sim} x$
 - $e^x - 1 \underset{x \rightarrow 0}{\sim} x$
 - pour tout $\alpha \in \mathbb{R}^*$, $(1+x)^\alpha - 1 \underset{x \rightarrow 0}{\sim} \alpha x$ et en particulier (pour $\alpha = \frac{1}{2}$), $\sqrt{1+x} - 1 \underset{x \rightarrow 0}{\sim} \frac{x}{2}$
 - $1 - \cos(x) \underset{x \rightarrow 0}{\sim} \frac{x^2}{2}$
- ⇒ Propriétés : transitivité, produit, quotient, puissance (pas de somme ni de composition!), théorème de substitution.

Applications linéaires

- ⇒ Définition d'une application linéaire, image du vecteur nul, vocabulaire (endomorphisme, isomorphisme, automorphisme)
- ⇒ Opérations : somme, multiplication par un scalaire, composition ([démonstration exigible](#)), réciproque d'applications linéaires.
- ⇒ Noyau : définition, sous-espace vectoriel de l'ensemble de départ ([démonstration exigible](#)), caractérisation de l'injectivité,
- ⇒ Image : définition, sous-espace vectoriel de l'ensemble d'arrivée, $Im(f) = \text{vect}(f(\vec{e}_1), \dots, f(\vec{e}_p))$ où $(\vec{e}_1, \dots, \vec{e}_p)$ base de l'ensemble de départ, caractérisation de la surjectivité.

Remarques aux colleurs

- Merci aussi de poser une petite question d'informatique (cf Annexe).

Exemples de programmes informatiques

Exercice 1

Ecrire en Python une fonction `existence` qui prend en entrée un tableau T et un nombre $element$ et renvoie `True` si $element$ se trouve dans le tableau T , `False` sinon.

```
def existence(T,element):
    a=len(T)      # nombre de lignes
    b=len(T[0])   # nombre de colonnes
    for i in range(a):      # parcours des lignes
        for j in range(b): # parcours des colonnes
            if T[i,j]==element: # on teste si T[i,j] est égal à élément
                return True
    return False # si on n' a pas trouvé element après avoir parcouru tout le tableau
```

Exercice 2

Ecrire en Python une fonction `MaximumTableau` qui prend en entrée un tableau T et renvoie la plus grande valeur de ce tableau

```
def MaximumTableau(T):
    a=len(T)      # nombre de lignes
    b=len(T[0])   # nombre de colonnes
    maxi = T[0,0] # initialisation avec la première valeur du tableau
    for i in range(a):      # parcours des lignes
        for j in range(b): # parcours des colonnes
            if T[i,j]>maxi: # on teste si T[i,j] est plus grand
                maxi=T[i,j] # on a trouvé une plus grande valeur
    return maxi
```

Exercice 3

Ecrire en Python une fonction `Moyenne` qui prend en entrée un tableau T et renvoie la moyenne de ses éléments :

```
def Moyenne(T):
    a=len(T)      # nombre de lignes
    b=len(T[0])   # nombre de colonnes
    S=0           # initialisation de la somme
    for i in range(a):      # parcours des lignes
        for j in range(b): # parcours des colonnes
            S=S+T[i,j]      # on rajoute l'élément T[i,j]
    return S/(a*b)         # formule pour la moyenne
```

Exercice 4

Ecrire une fonction `experience` qui prend en paramètre un entier n et simule n lancers successifs d'une pièce de monnaie équilibrée en renvoyant une liste aléatoire composée de n valeurs égales à 0 ou 1. On considérera que 0 correspond à Face et 1 à Pile.

```
from random import * # bibliothèque nécessaire pour créer des nombres aléatoires
def experience(n):
    L=[]              #liste vide initialement
    for i in range(n):      #on répète n fois
        L.append(randint(0,1)) # on rajoute 0 ou 1, choisi de manière aléatoire
    return L
```

Exercice 5

Créer une fonction `SommeLignes` prenant en entrée un tableau T et qui renvoie une liste contenant la somme de chaque lignes.

```
def SommeLignes(T):
    n=len(T)          # nombre de lignes
    p=len(T[0])      # nombre de colonnes
    L=[]             # la liste est vide au debut
    for i in range(n): # parcours des lignes
        S=0          # initialisation pour la somme de la ligne i
        for j in range(p): # parcours des colonnes
            S = S + T[i,j]
        L.append(S)    # on met la somme de la ligne i dans le liste L
    return L
```