Semaine n°17 du 02 au 06 février

Informatique(Python) : cf exemples en annexe

- boucle `while`, boucle `for`,.
- listes en Python : création d'une liste, extraction d'un élément, parcours d'une liste, concaténation, `len`, `append`...etc
- Graphique : module pyplot de mathplotlib, fonction plot et show. (Attention la fonction linespace (de numpy) n'est pas encoer connue).

Systèmes linéaires

- Définitions : système linéaire à n équations et p inconnues, solutions d'un système, systèmes équivalents, système compatible.
- Système échelonné : définition, méthode de résolution, rang d'un système échelonné, rang maximal, ensemble de solutions en fonction du rang, nombre de solutions d'un système échelonné.
- Méthode du pivot de Gauss pour échelonner un système.
- Rang d'un système linéaire quelconque, nombre de solutions d'un système linéaire.
- Système de Cramer : définition, un système de Cramer admet une unique solution.
- Exemples de résolution de système avec paramètre.

dénombrement

- Cardinal d'un ensemble : définition, cardinal d'un sous-ensemble, lien avec les ensembles de départ et d'arrivée des applications injectives, surjectives, bijectives (cas particulier d'une application ayant un ensemble de départ et d'arrivée de même cardinal).
- Cardinal d'une union : disjointe de deux ensembles, disjointe de n ensembles, union quelconque de deux ensembles.
- Cardinal d'un produit cartésien.
- p-liste sans répétition : définition, nombre de p-listes sans répétition d'un ensemble à n éléments.
- permutations : définition, nombre de permutations d'un ensemble.
- combinaisons : définition, nombre de p-combinaisons d'un ensemble à n éléments.
- cardinal de l'ensemble des parties d'un ensemble E fini (démonstration exigible).

Espaces vectoriels

- Espace vectoriel $\mathbb{K}^n$: vecteurs, saclaires, addition de deux vecteurs, multiplication d'un vecteur par un scalaire.
- Propriétés : Soient $\lambda \in \mathbb{K}$ et $\vec{u} \in \mathbb{K}^n$. Alors
  - $0 \cdot \vec{u} = \vec{0_{\mathbb{K}^n}}$
  - $\lambda \cdot \vec{0_{\mathbb{K}^n}} = \vec{0_{\mathbb{K}^n}}$
  - $\lambda \cdot \vec{u} = \vec{0_{\mathbb{K}^n}} \iff (\lambda = 0 \text{ ou } \vec{u} = \vec{0_{\mathbb{K}^n}})$
- Combinaison linéaire de vecteurs.
- Sous espace vectoriel de $\mathbb{K}^n$: partie de $\mathbb{K}^n$ contenant le vecteur nul et stable par combinaison linéaire.
- Intersection de deux sous-espaces vectoriels (démonstration exigible)
- Notation Vect$(\vec{u_1}, ...\vec{u_p})$ où $\vec{u_1}, \vec{u_p}$ sont des vecteurs de $\mathbb{K}^n$: ensemble des combinaisons linéaires des vecteurs $\vec{u_1}, ...\vec{u_p}$. C'est un sous espace vectoriel de $\mathbb{K}^n$ appelé le sous-espace vectoriel de $\mathbb{K}^n$ engendré par les vecteurs $\vec{u_1}, ...\vec{u_p}$.
- Différentes écritures d'un sous-espace vectoriel de $\mathbb{K}^n$:
  - Ecriture cartésienne : $A = \{(x, y, z) \in \mathbb{R}^3, x + y + z = 0 \text{ et } x - y = 0\}$.
  - Ecriture paramétrée : $A = \{(x, x, -2x), x \in \mathbb{R}\}$.
  - Ecriture sous forme d'une sous espace vectoriel engendré par une famille de vecteurs : A =vect $((1, 1, -2))$.

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 une liste L et un nombre *element* et renvoie `True` si *element* se trouve dans la liste L, `False` sinon.

```
def existence(L,element):
    n=len(L)   # taille de la liste
    for i in range(n):
        if L[i]==element:
            return True
    return False  # si on n' a pas trouvé element après avoir parcouru toute la liste
```

Exercice 2

Ecrire en Python une fonction `MaximumListe` qui prend en entrée une liste L et renvoie la plus grande valeur de cette liste

```
def MaximumListe(L):
    n=len(L)  #taille de la liste
    maxi=L[0]  #on considère temporairement que le max est le premier élément
    for i in range(n):
        if L[i]>maxi:
            maxi=L[i]  #on a trouvé une plus grande valeur
    return maxi
```

Exercice 3

Ecrire en Python une fonction `Somme` qui prend en entrée une liste L et renvoie la somme de ses éléments :

```
def Somme(L):
    n=len(L)   #taille de la liste
    S=0  #initialisation de la somme
    for i in range(n):
        S=S+L[i]
    return S
```

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):
        L.append(randint(0,1)) # 0 ou 1 choisi de manière aléatoire
    return L
```