

## Semaine n°16 du 29 janvier au 02 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
- ⇒ Chaîne de caractères.
- ⇒ Dictionnaire.

## Applications

- ⇒ Définitions : application, image, antécédent.
- ⇒ Application identité, application nulle, fonction indicatrice.
- ⇒ Image d'une partie de l'ensemble de départ pour une application  $f : E \rightarrow F$  :  

$$f(A) = \{f(x) | x \in A\}$$
- ⇒ Composition de deux applications.
- ⇒ Surjection : définition, méthode pour montrer qu'une application est surjective (en résolvant l'équation  $f(x) = y$ )
- ⇒ Injection : définition, méthode pour montrer qu'une application est injective ( $\forall (a, b) \in E^2, f(a) = f(b) \Rightarrow a = b$ )
- ⇒ bijection : définition, méthodes pour montrer qu'une application est bijective (en montrant qu'elle est injective et surjective ou en montrant que l'équation  $f(x) = y$  où  $y \in F$  admet une unique solution dans  $E$ , ou encore en utilisant le théorème de la bijection) ou qu'elle n'est pas bijective.
- ⇒ Application réciproque d'une bijection : définition, méthode pour trouver son expression, propriétés ( $f^{-1} \circ f = Id_E, f \circ f^{-1} = Id_F$ , symétrie des représentations graphiques par rapport à la droite d'équation  $y = x$ )
- ⇒ La composée de deux bijections est bijective et  $(g \circ f)^{-1} = f^{-1} \circ g^{-1}$ .

## 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.

**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
```