Programme de colle 22
Semaine du 30 mars 2026

Chapitre 19 : Systèmes linéaires

➢ **Cours à connaître**

- ✔ Définition système linéaire, coefficients du système, second membre, système homogène, systèmes équivalents.
- ✔ Système de Cramer
- ✔ Connaître les opérations élémentaires.
- ✔ Théorème sur le nombre de solutions d'un système.
- ✔ Matrice associée à un système linéaire
- ✔ Caractérisation de l'inversibilité d'une matrice : A est inversible ssi A est la matrice associée à un système de Cramer ; unicité solution de $AX = B$ est $X = A^{-1}B$.
- ✔ Réduite de Gauss et lien avec l'inversibilité.
- ✔ Méthode de Gauss-Jordan pour déterminer l'inverse d'une matrice

➢ **Exercices type**

- ✔ Résoudre un système à l'aide de la méthode du pivot de Gauss sous forme de système.
- ✔ Résoudre un système de Cramer homogène : il a pour unique solution le n-uplet $(0, \dots, 0)$.
- ✔ Déterminer si une matrice est inversible ou non et déterminer son inverse à l'aide de la méthode de Gauss-Jordan En déduire les solutions d'un système de Cramer associé.

Chapitre 20 : Théorie des graphes

➢ **Cours à connaître**

- ✔ Vocabulaire : Graphe orienté / non orienté, ordre d'un graphe, degré d'un sommet, sommets adjacents, graphe connexe, chemin / chaîne
- ✔ Lemme d'Euler ou des poignées de mains et son corollaire
- ✔ Matrice d'adjacence
- ✔ Théorème sur la matrice d'adjacence et le nombre de chemins
- ✔ Caractérisation de la connexité avec la matrice d'adjacence

➢ **Exercices type**

- ✔ Construire un graphe à partir d'un énoncé
- ✔ Utiliser le lemme d'Euler.
- ✔ Écrire la matrice d'adjacence d'un graphe / tracer un graphe en connaissant sa matrice d'adjacence.
- ✔ Déterminer le nombre de chemins de longueur d entre deux sommets donnés (utiliser les puissances de la matrice).
- ✔ Montrer qu'un graphe est connexe en utilisant la caractérisation avec la matrice d'adjacence.

Python : simulation d'expériences aléatoires

Il faut connaitre les commandes de base pour générer des nombres aléatoires :

- Importer la librairie Random : `import numpy.random as rd`
- Créer un nombre aléatoire de $[0;1[$: `rd.random()`
- Créer un vecteur de n nombres aléatoires de $[0;1[$: `rd.random(n)`
- Créer un entier aléatoire entre p et n : `rd.randint(p, n+1)` (attention : comme dans `range`, la dernière valeur n'est pas comptée). On peut aussi ajouter un paramètre pour avoir un vacteur avec plusieurs entiers, par exemple `rd.randint(1,4,10)` donne 10 entiers entre 1 et 3.

Il faut savoir les utiliser pour simuler une expérience aléatoire : lancer de dé, Pile ou Face, ou plusieurs expériences simultanées.

➢ Écrire une fonction `Piece` qui simule un lancer de pièce à Pile ou Face

```
import numpy.random as rd
def Piece() :
    alea = rd.random()
    if alea < 1/2 :
        return "Pile"
    else :
        return "Face"
```

Autre version :

```
import numpy.random as rd
def Piece() :
    alea = rd.randint(1,3) # entier aléatoire égale à 1 ou 2
    if alea == 1  :
        return "Pile"
    else :
        return "Face"
```

➢ Écrire une fonction `De` qui prend un entier n en paramètre et simule un lancer de n dés

```
import numpy.random as rd
def De(n) :
    alea = rd.randint(1,7,n)
    return alea
```