Semaine 26 : 12 au 16 mai 2025

A. Espaces vectoriels – révisions

B. Applications linéaires de $\mathbb{K}^p$ dans $\mathbb{K}^n$

Note aux interrogateurs : on étudie exclusivement les applications linéaires de $\mathbb{K}^p$ dans $\mathbb{K}^n$.

∗ définition, forme linéaires, endomorphismes; Opérations: somme, combinaison linéaire; $f(O_p) = O_n$; $f(-x) = -f(x)$.
∗ **Noyau d'une application linéaire:** définition; sev de $\mathbb{K}^p$; CNS d'injectivité.
∗ **Matrices et applications linéaires:** matrice qui représente une application linéaire dans deux bases fixées;
∗ **Image d'une application linéaire:** définition; sev de $\mathbb{K}^n$, CNS de surjectivité;
Soit $(u_1, \ldots, u_p)$ une base de $\mathbb{K}^p$ (pas forcément la base canonique!). Alors, $Imf = Vect(f(u_1), \ldots, f(u_p))$
∗ **Rang d'une application linéaire :** $rgf = \dim(Imf)$ (ne dépend pas du choix de la base); théorème du rang.
∗ **Matrices et applications linéaires:**
matrice qui représente une application linéaire dans deux bases fixées;
calcul matriciel de $f(x)$ $\forall x \in \mathbb{K}^p$; (cohérence des bases)
matrice de la somme, de λf, composée de deux applications linéaires (cohérence des bases)
matrice de f^{-1}

Capacités exigibles :

1. Montrer qu'une application est linéaire.
2. former la matrice d'une application linéaire dans deux bases fixées (pas forcément canoniques)
3. calculer le noyau, l'image, le rang d'une application linéaire.
4. Utilisation de matrices pour la bijectivité, le calcul de f^{-1} et les opérations (polynômes d'endomorphismes par exemple).

C. Langage `python`

Simulation d'expériences aléatoires, marches aléatoires (import random as rd) :
fonctions `choice(L)`, `L.pop()`, `L.pop(k)`, `shuffle(L)` où L est une liste de nombres ;
fonctions `randrange(a,b)`, `randint(a,b)` où a et b sont des entiers avec $a < b$;
fonction `random()`

Déroulement de la colle :

La colle commence par une question d'informatique (langage `python`) parmi :

1. Tri par insertion : algorithme et explications sur un exemple.
2. Tri par sélection *en place* : algorithme et explications sur un exemple.
3. Tri par sélection *non en place* : algorithme et explications sur un exemple.
4. On tire avec remise cinq boules d'une urne contenant dix boules numérotées de 1 à 10. Écrire un programme qui renvoie le maximum des cinq numéros obtenus.
5. Soit N un entier supérieur à 10. Une urne contient N boules numérotées de 1 à N. On tire au hasard et simultanément 10 boules de l'urne. Écrire un programme qui renvoie le minimum des dix numéros obtenus.
6. Lors d'un jeu, un candidat choisit une question en tirant au hasard un papier parmi trois. Il y a:

   • une question facile, pour laquelle on a 3 chances sur 4 de donner la réponse exacte,

   • une question moyenne, pour laquelle on a 2 chances sur 5 de donner la réponse exacte,

   • une question difficile, pour laquelle on a 1 chances sur 5 de donner la réponse exacte.

   Écrire une fonction `reponse()` qui renvoie `True` si le candidat donne la bonne réponse, `False` sinon.
   on pourra une variable qui vaut `0,1,2` *suivant le niveau de la question.*
7. Un mobile se déplace aléatoirement le long d'un axe horizontal d'origine 0, sur des points à coordonnées entières, positives ou nulles.
   Les déplacements sont effectués selon le protocole suivant:

   • à l'instant 0, le mobile est sur l'origine O d'abscisse 0.

   • si, pour tout entier naturel n, le mobile se trouve à l'instant n sur le point d'abscisse k $(0 \leqslant k \leqslant n)$, alors il sera à l'instant $n+1$, soit sur le point d'abscisse $k+1$ avec la probabilité $\dfrac{1}{3}$, soit sur le point O avec la probabilité $\dfrac{2}{3}$.

   Écrire un programme qui simule cette marche aléatoire et renvoie l'abscisse du mobile à l'instant n.

Puis passage aux exercices