Lycée Hoche
ECG1 A

2024-2025
Mathématiques

Devoir surveillé numéro 3

*Devoir du **21/12/2024**.*

L'usage de documents de cours et d'appareils électroniques est interdit. Le soin de la copie et la qualité de la rédaction seront pris en compte de manière importante dans la notation. Les résultats démontrés non encadrés pourront être ignorés par le correcteur. On rappelle que la qualité de l'argumentation et la rigueur (donc le soin accordé aux détails) sont au cœur des mathématiques. Sauf mention explicite du contraire, tout résultat demandé doit être démontré.

Bon courage à toutes et à tous!

Exercice 1

1. (*a*) Soient n et l deux réels. On pose $A = \{(j,x) \in \mathbb{R}^2 | j + x = n + l\}$ et $B = \{(n-o, o+l) | o \in \mathbb{R}\}$. Montrer que $A = B$.

   (*b*) La fonction Joyeuxnoel : $\left|\begin{array}{ccc} \mathbb{R}^2 & \longrightarrow & \mathbb{R} \\ (j,x) & \longmapsto & j+x \end{array}\right.$ est-elle injective?

   (*c*) Est-elle surjective ?

2. On pose $A = \{P \in \mathbb{R}[X] | P(0) = 0\}$ et $B = \{XQ(X) | Q \in \mathbb{R}[X]\}$. Montrer que $A = B$.

3. Vrai ou faux? Justifier soigneusement.

   (*a*) $f : \left|\begin{array}{ccc} \mathbb{R}_+ & \longrightarrow & \mathbb{R}_+ \\ x & \longmapsto & x^2+1 \end{array}\right.$ est bijective.

   (*b*) $g : \left|\begin{array}{ccc} \mathbb{R}[X] & \longrightarrow & \mathbb{R}[X] \\ P & \longmapsto & P' \end{array}\right.$ est injective.

   (*c*) $h : \left|\begin{array}{ccc} \mathbb{R}^3 & \longrightarrow & \mathbb{R}^3 \\ (a,b,c) & \longmapsto & (c,b,a) \end{array}\right.$ est bijective.

4. Déterminer le terme général de la suite u définie par $u_0 = 1$, $u_1 = 3$ et $\forall n \in \mathbb{N}, u_{n+2} = 2\sqrt{2}u_{n+1} - 2u_n$.

5. Soit r un réel et soit P un polynôme. Montrer que r est racine de P si et seulement si $X - r | P$.

6. Soit λ un réel. Résoudre, en fonction de λ, le système $(S) : \begin{cases} x - 2y + z = 1 \\ -x - y + z = 2 \\ x - 5y + \lambda z = 6 \end{cases}$ d'inconnues x, y et z.

7. Soit E un ensemble. Soient A, B et C des parties de E deux à deux disjointes telles que $A \cup B \cup C = E$.

   Montrer que pour toute partie P de E, $P = (P \cap A) \cup (P \cap B) \cup (P \cap C)$ et que $P \cap A$, $P \cap B$ et $P \cap C$ sont deux à deux disjoints.

8. Énoncer le théorème de croissance comparée.

9. Soit $(u_n)_{n\in\mathbb{N}}$ une suite bornée et $(v_n)_{n\in\mathbb{N}}$ une suite telle que $v_n \xrightarrow[n\to+\infty]{} 0$.

   Montrer que $u_n v_n \xrightarrow[n\to+\infty]{} 0$.

10. Écrire un code Python permettant de tracer les 100 premiers termes de la suite u définie par $u_0 = 1$ et $\forall n \in \mathbb{N}, u_{n+1} = nu_n + 1$.

Exercice 2 *Quelques questions indépendantes de calcul*

1. Montrer que la suite u donnée par le terme général suivant a une limite et la calculer. On admet que ces suites sont définies sur $\mathbb{N}^*$.

   (a) $u_n = \dfrac{2n + n^2 + 2^n}{3n - n^3 - 3^n}$. (b) $u_n = \ln(n^2 + \mathrm{e}^{-n}) - n\ln(n)$. (c) $u_n = \dfrac{e^{\sqrt{n}}}{n^2}$.

2. Factoriser au maximum le polynôme $P(X) = X^5 + \dfrac{3}{2}X^4 - \dfrac{1}{2}X^2$.

3. On considère la fonction $f : \mathbb{R}_+^* \to \mathbb{R}$ donnée par $f(x) = (x^x)^x$.

   (a) Justifier que f est correctement définie.

   (b) Montrer que f est dérivable et calculer f'.

   (c) Dresser le tableau de variations de f, en ignorant les limites.

   (d) On pose $x_n = \dfrac{1}{n}$ pour tout $n \in \mathbb{N}^*$ (*pour de simples raisons typographiques*).

   Déterminer la limite de $((x_n^{x_n})^{x_n})_n$.

   (e) Déterminer la limite de $(x_n^{(x_n^{x_n})})_n$.

4. On pose, pour tout $n \in \mathbb{N}^*$:

$$S_n = \sum_{k=1}^{n}(k-1)^2 k!,\ T_n = \sum_{k=1}^{n}(k^2+1)k! \text{ et } W_n = \sum_{k=1}^{n} k \times k!.$$

   (a) Montrer $\forall k \in \mathbb{N}^*, (k+1)! - k! = k \times k!$.

   (b) En déduire le calcul de W_n, pour tout $n \in \mathbb{N}^*$.

   (c) Montrer que : $\forall n \in \mathbb{N}^*, S_n = 2 + (n-2)(n+1)!$.

   (d) Exprimer S_n en fonction de T_n et de W_n, pour tout $n \in \mathbb{N}^*$.

   (e) En déduire le calcul explicite de T_n en fonction de $n \in \mathbb{N}^*$.

5. On lance trois fois un dé classique à 6 faces et on note, dans l'ordre, les faces obtenues.

   (a) Décrire mathématiquement l'ensemble des résultats possibles et donner son cardinal.

   (b) Combien de résultats possibles contiennent exactement les faces 2, 4 et 6 ?

   (c) Combien de résultats possibles font apparaitre (au moins) une face paire et (au mois) une face impaire ?

Exercice 3

Le but de cet exercice est de s'intéresser à une suite définie de manière implicite, traitée en troisième partie. Les deux premières parties établissent des résultats nécessaires à cette étude.

Partie I : Une croissance comparée.

Le but de cet exercice est de déterminer la limite de la suite $(\frac{a^n}{n!})_{n\in\mathbb{N}}$, où a est un réel positif.

On fixe donc, pour cette partie, un réel $a \geq 0$ et on pose, pour tout $n \in \mathbb{N}, u_n = \frac{a^n}{n!}$.

1. On suppose $a \leq 1$. Déterminer la limite de u_n.
2. On suppose maintenant $a > 1$.
   (a) Montrer qu'il existe un entier naturel n_0 tel que : $\forall n \geq n_0, u_{n+1} < \frac{1}{2}u_n$. On fixe un tel entier n_0 pour la fin de cette partie.
   (b) Montrer que : $\forall n \geq n_0, u_n \leq \frac{1}{2^{n-n_0}}u_{n_0}$.
   (c) En déduire la limite de $(u_n)_{n\in\mathbb{N}}$.

Partie II : Étude de fonctions.

On pose, pour tout $n \in \mathbb{N}^*$, $f_n(x) = \frac{x^n}{n!} - \sum_{k=0}^{n-1} \frac{x^k}{k!}$.

3. Soit $n \in \mathbb{N}^*$. Justifier que l'on définit ainsi une fonction $f_n : \mathbb{R} \to \mathbb{R}$ dérivable, et déterminer une relation entre f'_{n+1} et f_n.

   On admet dans la suite que $f_n(x) \xrightarrow[x\to+\infty]{} +\infty$.

4. (a) Écrire un code définissant une fonction Python d'entête `def f(n,x):` prenant en entrée un entier `n` ≥ 1 et un réel `x` et renvoyant en sortie $f_{\texttt{n}}(\texttt{x})$.
   (b) Écrire un code Python permettant de tracer f_{10} sur $[0, 5]$.
5. Étudier la fonction f_1 sur $\mathbb{R}_+$. On déterminera ses variations et ses points d'annulation.
6. Montrer par récurrence que pour tout $n \in \mathbb{N}^*$:
   - Il existe un unique réel positif, noté x_n dans la suite, tel que $f_n(x_n) = 0$, et
   - f_n est strictement négative sur $[0, x_n[$ et strictement positive sur $]x_n, +\infty[$.
7. Établir, pour tout $n \in \mathbb{N}^*$, le tableau de variation de f_n sur $\mathbb{R}_+$ et justifier $x_n > 0$.
8. A quelle condition sur $n \in \mathbb{N}^*$ la fonction f_n est-elle injective sur $\mathbb{R}_+$?

Partie III : Étude de la suite $(x_n)_{n\in\mathbb{N}^*}$.

9. Montrer que $(x_n)_{n\in\mathbb{N}^*}$ est strictement croissante. Que dire de la nature de cette suite ?
10. Montrer que la suite v définie par $\forall n \in \mathbb{N}^*, v_n = \sum_{k=0}^{n-1} \frac{x_n^k}{k!}$ est strictement croissante et à termes positifs.
11. On suppose pour ces deux sous-questions que $(x_n)_n$ est majorée.
    (a) Montrer que $\frac{x_n^n}{n!} \xrightarrow[n\to+\infty]{} 0$.
    (b) En déduire que $v_n \xrightarrow[n\to+\infty]{} 0$.
12. Déduire de la question précédente la limite de $(x_n)_n$.

Exercice 4 *Étude d'une suite récurrente*

On considère la fonction $f :]0, +\infty[\to \mathbb{R}$ définie par :

$$\forall x \in]0, +\infty[, f(x) = \frac{\mathrm{e}^{-x}}{x}.$$

On considère également la suite $(u_n)_{n\in\mathbb{N}}$ définie par $u_0 = 1$ et $\forall n \in \mathbb{N}, u_{n+1} = f(u_n)$.

1. (a) Dresser le tableau de variation complet de f. Limites admises : $f(x) \xrightarrow[x\to 0]{} +\infty$ et $f(x) \xrightarrow[x\to+\infty]{} 0$.

   (b) Vérifier que pour tout $n \in \mathbb{N}$, u_n est bien défini et strictement positif.

2. (a) Écrire une fonction Python d'entête `def rangSup(a)` prenant en entrée un réel `a` > 0 et renvoyant en sortie le plus petit entier n tel que $u_n > a$.

   (b) De même, on a écrit une fonction `rangInf` prenant en entrée un réel `a` > 0 et renvoyant en sortie le plus petit entier n tel que $u_n < a$. On utilise ces fonctions dans l'interpréteur, ce qui fournit le résultat ci-contre.

```
>>> rangSup(10**5)
6

>>> rangSup(10**7)
8

>>> rangInf(10**-5)
5

>>> rangInf(10**-7)
7
```

   Commenter le résultat obtenu. On pourra émettre une conjecture sur le comportement de $(u_n)_n$ (croissance, caractère majoré ou non, caractère convergent ou non).

3. On définit la fonction g sur $[0, +\infty[$ par $g(x) = \mathrm{e}^{-x} - x^2$.

   (a) Dresser le tableau de variation de g. Limite admise : $g(x) \xrightarrow[x\to+\infty]{} -\infty$.

   (b) Montrer que g réalise une bijection de $[0, +\infty[$ vers $]-\infty, 1]$.

   (c) En déduire que l'équation $f(x) = x$ possède une unique solution sur $]0, +\infty[$, que l'on notera α.

   (d) Justifier $\mathrm{e}^{-1} < \alpha < 1$.

4. On considère la fonction h définie par $h(x) = f \circ f(x)$.

   (a) Justifier que h a pour domaine $]0, +\infty[$. Quelle est la monotonie de h?

   (b) Montrer que $u_2 > u_0$.

   (c) En déduire que la suite v donnée par $v_n = u_{2n}$ (pour tout $n \in \mathbb{N}$) est strictement croissante. On pourra expliciter une relation de récurrence vérifiée par v.

   (d) Montrer de même que $w = (u_{2n+1})_{n\in\mathbb{N}}$ est strictement décroissante.

   (e) En déduire que $(u_{2n+1})_{n\in\mathbb{N}}$ converge vers un réel $l \in [0, \mathrm{e}^{-1}]$.

5. (a) Déterminer, pour tout $x > 0$, une expression de $h(x)$ en fonction de $g(x)$.

   (b) Résoudre alors l'équation $h(x) = x$ d'inconnue $x \in]0, +\infty[$.

   (c) En déduire que $u_{2n+1} \xrightarrow[n\to+\infty]{} 0$.

   (d) En déduire que $u_{2n} \xrightarrow[n\to+\infty]{} +\infty$.

6. Qu'en déduire sur la limite de $(u_n)_n$?

Exercice 5

Programmation d'un jeu de cartes. Une annexe à consulter présente quelques rappels informatiques.

Le but de cet exercice est de coder quelques fonctions liées à la représentation informatique de jeux de cartes. Dans cet exercice, on prendra comme exemple des cartes issues d'un jeu de cartes classique. Les trois parties de cet exercice se suivent.

On dispose de cartes, chacune ayant les caractéristiques suivantes :

- Une **famille**. Dans cet exercice, il y en aura quatre : Carreau, Coeur, Trèfle, Pique.
- Une **hauteur** qui est un nombre entier. Dans cet exercice, il y en aura 13 : de 1 (pour l'As) à 13 (pour le Roi).

Un **paquet de cartes** est donné par un certain nombre de ces cartes dans un certain ordre, sans répétitions de cartes.

Une **main de n cartes** d'un paquet est donnée par n cartes qu'on peut tirer de ce paquet (donc distinctes), sans prendre en compte l'ordre (où n est un entier naturel).

Pour la **représentation informatique** de ces objets, on adopte les conventions suivante :

- Une **carte** est donnée par un couple `(f,h)` où `f` est une chaîne de caractère représentant la famille de cette carte, et `h` sa hauteur.

  Dans cet exercice, les familles sont représentées par les chaîne de caractères indiquées ci-dessous.

Famille	Carreau	Coeur	Trèfle	Pique
Code	`"Ca"`	`"C"`	`"T"`	`"P"`

- Un **paquet de cartes** est représenté par une liste de cartes.

  Par exemple, la liste `[ ("C",2), ("T",12) ]` désigne le paquet contenant le 2 de Coeur, puis la Dame de Trèfle. On pensera alors au 2 de Coeur comme à la carte du dessus de ce paquet.

- Une **main** est représentée par la liste des cartes qu'elle contient, il existe donc plusieurs représentations informatiques possibles d'une même main (on peut permuter les cartes de cette liste sans changer la main considérée).

Bien-sûr, toute fonction définie dans une question pourra être utilisée dans les question suivante.

Partie I : Quelques commandes générales

1. Recopier et compléter le code Python suivant pour qu'à la fin de son exécution, la variable `PaquetClassique` contienne un paquet classique de 52 cartes (avec une fois chaque carte de chaque famille et de chaque hauteur).

```
PaquetClassique=[]
for f in ["C", "Ca", "T", "P"]:
    for h in ... :
        PaquetClassique.append( ... )
```

2. Écrire le code d'une fonction Python d'entête `def MainsEgales(M1,M2):` prenant en entrée deux listes représentant des mains et renvoyant en sortie `True` si ces listes représentent la même main, et `False` sinon. *Indication : deux listes représentent la même main si et seulement si elles ont la même taille et toute carte de la première liste est dans la seconde liste.*

3. Écrire le code d'une fonction Python d'entête `def Famille(f,Pa):` prenant en entrée une famille `f` et un paquet `Pa` et renvoyant en sortie le paquet formé des cartes de `Pa` dont la famille est `f`, dans le même ordre.

   Par exemple, `Famille("C", [ ("C",2), ("T",12), ("C",12) ])` devra renvoyer `[ ("C",2), ("C",12)]`.

4. En déduire le code d'une fonction d'entête `def MainCoeur(Paquet):` prenant en entrée un paquet de cartes et renvoyant en sortie la liste des mains de 2 cartes de ce paquet contenant exactement une carte de coeur (sans répétitions de mains).

Partie II : Jeu à points

On dit maintenant que chaque carte vaut un certain nombre de points. Le nombre de points d'une carte est le produit de sa hauteur par un multiplicateur dépendant de sa famille, selon le tableau suivant.

Famille	Carreau	Coeur	Trèfle	Pique
Multiplicateur	1	4	2	3

Par exemple, l'As de coeur vaut $4 \times 1 = 4$ points tandis que la dame de Pique vaut $3 \times 12 = 36$ points.

5. Écrire le code d'une fonction `Points` prenant en entrée une carte et renvoyant en sortie le nombre de points de cette carte.
6. Écrire le code d'une fonction d'entête `def PointsTotaux(Paquet):` prenant en entrée un paquet de carte et renvoyant en sortie la somme des points de toutes les cartes de ce paquet.
7. Écrire le code d'une fonction `TriPoints` prenant en entrée un paquet et triant ce paquet de sorte que les cartes apparaissent par nombre de points décroissants.

Partie III : Simulation d'une simple bataille

Une bataille se déroule de la manière suivante :

- On considère un paquet de cartes ayant un nombre paire de cartes. On mélange ce paquet de cartes.
- On forme deux paquets avec ce paquet (le paquet du joueur 1 et le paquet du joueur 2), chacun ayant la moitié du nombre total de cartes dans le paquet initial.
- On procède à des manches : on tire la première carte des paquets de chaque joueurs. Le joueur ayant la carte avec le plus de points gagne un cookie sur cette manche, et aucun cookie n'est gagné en cas d'égalité. Les cartes tirées sont retirées des paquets des joueurs.
- On répète les manches précédentes jusqu'à avoir tiré toutes les cartes des joueurs. Le joueur avec le plus de cookies gagne la bataille, et on déclare l'égalité s'ils ont le même nombre de cookies.

8. Écrire le code d'une fonction d'entête `def Mélange(Paquet)` prenant en entrée un paquet `Paquet` et mélangeant *suffisamment* ce paquet. *On utilisera la fonction* `rd.randint` *présentée en annexe. Votre fonction devra changer aléatoirement des cartes de place dans le paquet un grand nombre de fois de sorte que toute carte ait une chance de se retrouver à toute place à la fin.*
9. Recopier et compléter le code suivant pour qu'il définisse une fonction `TireCarte` prenant en entrée un paquet non vide `P` et renvoyant en sortie la première carte `c` du paquet et modifiant au passage le paquet `P` pour lui retirer sa première carte.

```
def TireCarte(P):
    c=...
    ...  # retrait de la première carte de P
    return(c)
```

10. Écrire le code d'une fonction d'entête `def SimuleBataille(Paquet):` prenant en entrée un paquet avec un nombre pair de cartes, et qui :
    - Simule une bataille avec ce paquet selon les règles ci-dessus,
    - **affiche** après chaque manche le nombre de cookies de chaque joueur ("Joueur 1 : ... Cookies, Joueur 2 : ... Cookies"),
    - fini par **afficher** "Le joueur 1 a gagné", "le joueur 2 a gagné" ou " Égalité" en fonction du résultat.

Annexe informatique

Si la variable `u` contient un couple `(a,b)`, la commande `u[0]` renvoie `a` et `u[1]` renvoie `b`.

Si `L` est une liste :

- La commande `a in L` renvoie `True` si `a` est un élément de `L`, et `False` sinon.
- `L[i:j]` renvoie la sous-liste de `L` de d'indice `i` à l'indice `j-1` (inclus), pour `i` et `j` convenables.
- `del L[i]` retire l'élément en position `i` de la liste `L`, pour tout `i` convenable.

Si le module `numpy.random` a été importé avec la commande `import numpy.random as rd`, chaque appel de la commande

`rd.randint(a,b)`

renvoie un nombre entier aléatoire nouvellement choisi entre `a` et `b-1`, où `a` et `b` sont des entiers.

Par exemple, `rd.randint(0,5)` renvoi un nombre entier aléatoire entre 0 et 4, et un second appel de cette commande tirera à nouveau un entier aléatoire entre 0 et 4 indépendamment du précédent.

— Fin de l'énoncé —