Devoir maison n° 9
MP
pour jeudi 15 janvier 2026

La présentation, la lisibilité, l'orthographe, la clarté et la précision des raisonnements entrent pour une part importante dans l'appréciation des copies. Il faut souligner ou encadrer les résultats.
Bon travail !

✍ : proche du cours, méthode à connaître, exercice de structure classique... Les exercices ✍ sont OBLIGATOIRES.
↯ : problème de concours, exercice pour chercher plus... Les exercices ↯ sont facultatifs, mais bien évidemment CONSEILLÉS.

Exercice 1 – ✍

On dipose d'une pièce truquée. La probabilité qu'elle tombe sur Pile est $\frac{1}{3}$ et la probabilité qu'elle tombe sur Face est $\frac{2}{3}$. On considère une succession de lancers de cette pièce. Pour $n \in \mathbb{N}$, on note A_n l'événement :

$A_n =$ « il a fallu attendre $n+3$ lancers pour observer pour la première fois 3 Face consécutifs »

et on note a_n la probabilité de A_n. Pour $n \in \mathbb{N}^*$, on note B_n l'événement :

$B_n =$ « aucune séquence de Face de longueur 3 n'apparaît dans la suite des n premiers lancers »

et on note b_n la probabilité de B_n.
On se propose de montrer, de deux façons différentes, qu'avec une probabilité 1, le nombre de lancers nécessaires pour obtenir 3 Face consécutifs est fini, ou encore : la probabilité que l'on n'observe jamais 3 Face consécutifs est nulle.

1. Expliciter A_0, A_1 et A_2 et donner leurs probabilités.
2. Calculer b_1, b_2 et b_3. Montrer qu'il existe des réels α_1, α_2 et α_3, que l'on déterminera, tels que :

$$\forall n \geqslant 4,\ b_n = \alpha_1 b_{n-1} + \alpha_2 b_{n-2} + \alpha_3 b_{n-3} \qquad (*)$$

Première méthode.

3. Montrer, par un argument probabiliste, que la suite $(b_n)_{n\geqslant 1}$ est décroissante.
4. Montrer que :

$$\forall n \geqslant 3 \quad 1 - b_n = \sum_{k=0}^{n-3} a_k$$

5. En déduire la convergence de la série $\sum a_k$ et sa somme $\sum_{k=0}^{\infty} a_k$.

Deuxième méthode.

6. Montrer, par un argument probabiliste, que la série de terme général a_n converge.

7. Montrer que :
$$\forall n \geqslant 2 \quad a_n = \frac{8}{81} b_{n-1}$$
8. En déduire que la série $\sum b_k$ converge et donner sa somme $\sum_{k=1}^{\infty} b_k$. On pourra utiliser la relation (*).
9. En déduire la valeur de la somme $\sum_{k=0}^{\infty} a_k$.
10. Conclure quant à la probabilité de ne jamais observer trois Face consécutifs.

Exercice 2 – ⚡

Trois preuves d'une égalité combinatoire

A/ Avec la fonction Beta

On pose, pour tout couple (n,p) d'entiers naturels, $B(n,p) = \displaystyle\int_0^1 t^n (1-t)^p \, \mathrm{d}t$.

1. Relier, pour tout couple $(n,p) \in (\mathbb{N}^*)^2$, les réels $B(n,p)$ et $B(n+1,p-1)$. En déduire, pour tout couple $(n,p) \in \mathbb{N}^2$, l'égalité : $B(n,p) = \dfrac{n!\,p!}{(n+p+1)!}$.
2. Soit $(m,n) \in \mathbb{N}^* \times \mathbb{N}$. Établir l'égalité
$$(\mathcal{E}) \quad \frac{(m-1)!\,n!}{(m+n)!} = \sum_{k=0}^{n} \binom{n}{k} \frac{(-1)^k}{m+k}.$$

B/ Avec des différences finies

On note Δ l'application qui, à chaque suite réelle $u = (u_p)_{p\in\mathbb{N}^*}$, associe la suite
$$\Delta(u) = (u_{p+1} - u_p)_{p\in\mathbb{N}^*}.$$

On note $\Delta^2 = \Delta \circ \Delta$, $\Delta^3 = \Delta \circ \Delta \circ \Delta$, etc., et $\Delta^0 = \mathrm{Id}_{\mathbb{R}^{\mathbb{N}^*}}$.

3. Prouver, pour toute suite $u = (u_p)_{p\in\mathbb{N}^*}$, pour tout entier naturel n et tout entier $p \in \mathbb{N}^*$, l'égalité
$$(\Delta^n(u))_p = \sum_{k=0}^{n} \binom{n}{k} (-1)^{n-k} u_{p+k}$$
4. En utilisant la suite $u = \left(\dfrac{1}{p}\right)_{p\in\mathbb{N}^*}$ établir à nouveau l'égalité $(\mathcal{E})$.

C/ Par un raisonnement probabiliste

Si $n \in \mathbb{N}^*$ et $A_1, A_2, \ldots, A_n$ sont des événements d'un espace probabilisé $(\Omega, \mathcal{T}, \mathrm{P})$, on dispose de la formule suivante, appelée *formule du crible*, et qui n'est pas au programme :
$$\mathrm{P}\left(\bigcup_{k=1}^{n} A_k\right) = \sum_{k=1}^{n} (-1)^{k-1} \sum_{1\leqslant i_1 < i_2 < \cdots < i_k \leqslant n} \mathrm{P}\left(A_{i_1} \cap A_{i_2} \cap \cdots \cap A_{i_k}\right)$$

5. Soit $A_1, A_2, \ldots, A_n$ et B des événements d'un espace probabilisé $(\Omega, \mathcal{T}, \mathrm{P})$. Établir l'égalité

$$\mathrm{P}\left(B \cap \bigcap_{i=1}^{n} A_i\right) = \mathrm{P}(B) + \sum_{k=1}^{n} (-1)^k \sum_{1 \leqslant i_1 < i_2 < \cdots < i_k \leqslant n} \mathrm{P}\left(B \cap \overline{A_{i_1}} \cap \overline{A_{i_2}} \cap \cdots \cap \overline{A_{i_k}}\right)$$

Soit $(m, n) \in (\mathbb{N}^*)^2$. On considère une urne contenant une boule noire notée N, n boules blanches notées $B_1, B_2, \ldots, B_n$ et $m-1$ boules rouges notées $R_1, R_2, \ldots, R_{m-1}$. On effectue des tirages successifs d'une boule dans cette urne, sans remise de la boule dans l'urne après tirage, jusqu'à épuisement de l'urne. Un résultat de cette expérience aléatoire est donc une $(m+n)$-liste (ordonnée...) composée des $(m+n)$ symboles $B_1, B_2, \ldots, B_n$, $R_1, R_2, \ldots, R_{m-1}$, et N. On note B l'événement constitué des tirages où la boule noire est tirée avant chacune des boules rouges, et, pour tout entier $i \in [\![1, n]\!]$, A_i l'événement constitué des tirages où la boule noire est tirée après la boule blanche B_i.

Les questions qui suivent sont plus difficiles, uniquement pour les étudiants à l'aise.

6. Donner la probabilité $\mathrm{P}(B)$ (on recommande de s'y prendre sans calcul).
7. Calculer la probabilité $\mathrm{P}\left(B \cap \bigcap_{i=1}^{n} A_i\right)$.
8. Prouver, pour tout entier $k \in [\![1, n]\!]$ et tout k-liste $(i_1, i_2, \ldots, i_k)$ avec $1 \leqslant i_1 < i_2 < \ldots < i_k \leqslant n$, l'égalité
$$\mathrm{P}\left(B \cap \overline{A_{i_1}} \cap \overline{A_{i_2}} \cap \cdots \cap \overline{A_{i_k}}\right) = \frac{1}{m+k}$$
9. (a) Soit $k \in [\![1, n]\!]$. Déterminer —en le justifiant— le nombre de k-listes $(i_1, i_2, \ldots, i_k)$ d'entiers telles que $1 \leqslant i_1 < i_2 < \cdots < i_k \leqslant n$.
   (b) En déduire à nouveau l'égalité $(\mathcal{E})$.

Exercice 3 – ✍

Un appendice, situé à la fin du problème, rappelle quelques fonctions du langage Python et, pour tout ce qui ressort de la simulation de l'aléatoire, de la bibliothèque `numpy.random`.

Soit $(X_n)_{n \geqslant 0}$ une suite de variables aléatoires mutuellement indépendantes, de même loi

$$P(X_i = +1) = p \quad \text{et} \quad P(X_i = -1) = 1 - p$$

où p est un réel vérifiant $0 < p < 1$.

Si s est un entier relatif, on définit la marche aléatoire d'origine p comme la suite $(S_n)_{n \geqslant 0}$ de variables aléatoires définies par $S_0 = s$ et, pour tout $n \geqslant 1$:

$$S_n = s + X_1 + \cdots + X_n \quad (*)$$

l'entier n étant identifiable à un temps discrétisé. Autrement dit, S_n est la position d'un marcheur partant de la position initiale s et effectuant n pas successifs, chaque pas étant, de manière indépendante :

— un pas vers la droite avec une probabilité p,

— un pas vers la gauche avec une probabilité $1 - p$.

Dans toute la suite, s est un entier strictement positif, et on s'intéresse à la valeur du premier instant de sortie de l'intervalle entier $[\![1; b-1]\!]$, c'est-à-dire du plus petit entier n tel que $S_n \in \{0, b\}$.

Le but de cette partie est de fournir une modélisation informatique d'une telle marche aléatoire, ainsi que des fonctions Python permettant de déterminer numériquement le temps de sortie d'un marcheur, en fonction de la position s d'origine de la marche.

1. Écrire une fonction `Pas(p)` modélisant une variable X_i, c'est-à-dire une fonction renvoyant la valeur $+1$ avec la probabilité p, et la valeur -1 avec la probabilité $1-p$.
2. Écrire une fonction `Marche(s, n, p)` prenant en paramètres deux entiers s et n, et le réel p supposé appartenir à l'intervalle $]0;1[$, et retournant une liste $(S_k)_{0\leqslant k\leqslant n}$ telle que définie par (*).
3. On se fixe deux entiers $2 \leqslant s < b$ tels que $0 < s < b$. On s'intéresse au temps de sortie d'un marcheur initialement en s, défini comme le nombre de pas nécessaires pour sortir de l'intervalle $[\![1; b-1]\!]$. *On admet que, avec une probabilité 1, ce temps de sortie est fini, et qu'il admet de plus une espérance.*
   Écrire une fonction `TempsSortie(s, b, p)` retournant ce temps de sortie.
4. Écrire une fonction `TempsMoyen(s, b, p, N)` retournant le temps de sortie du marcheur aléatoire, moyenné sur N expériences.

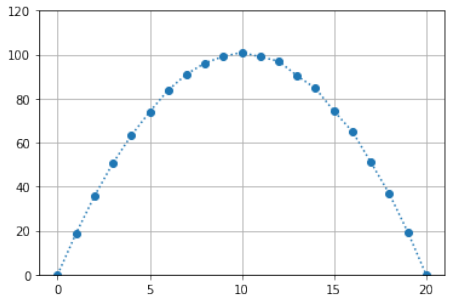
On s'intéresse désormais à la probabilité de sortie du côté positif, c'est-à-dire la probabilité que le marcheur passe par l'abscisse b avant de passer (éventuellement) par 0.

5. Écrire une fonction `SortiePositive(s, b, p)` retournant la valeur 1 si le marcheur sort par le côté positif, et 0 s'il sort par le côté négatif.
6. Écrire une fonction `ProbaSortiePositive(s, b, p, N)` estimant la probabilité de sortie du côté positif par un décompte moyen sur N expériences.
7. On crée des listes de nombres avec les instructions suivantes :

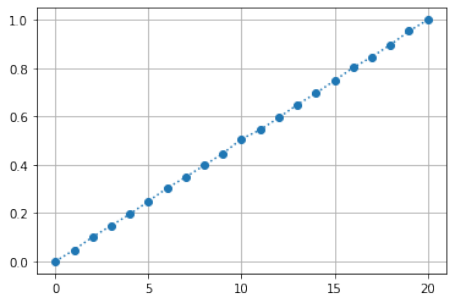
```
S, T, P = [], [], []
b = 20
for s in range(b+1):
    S.append(s)
    T.append(TempsMoyen(s, b, .5, 10000))
    P.append(ProbaSortiePositive(s, b, .5, 10000))
```

On fait maintenant exécuter les deux instructions suivantes, dont les sorties graphiques sont présentées sous chaque instruction :

`plt.plot(S, T)`



`plt.plot(S, P)`



Expliquer ces résultats : allure, valeurs spécifiques.

Appendice Python

On utilise la bibilothèque `random` de la librairie `numpy` : `import numpy.random as rd`
On a alors accès aux fonctions suivantes :

Instruction	Valeur retournée
`rd.random()`	un réel (flottant) de $[0;1[$ avec probabilité uniforme
`rd.randint(a, b)`	un entier de $\{a, a+1, \ldots, b-1\}$avec probabilité uniforme
`rd.randint(a, b, n)`	un tableau de n entiers de $\{a, a+1, \ldots, b-1\}$ avec probabilité uniforme
`rd.binomial(n, p)`	un entier de $[\![0, n]\!]$ selon la loi binomiale $\mathcal{B}(n,p)$
`rd.binomial(n, p, k)`	un tableau de k entiers de $[\![0, n]\!]$ selon la loi binomiale $\mathcal{B}(n,p)$