Devoir surveillé 4

Samedi 21 janvier 2023 - 4h00

La qualité de la rédaction, le soin porté à la copie, la lisibilité, l'orthographe, la rigueur du vocabulaire ainsi que la clarté des raisonnements sont des critères importants d'évaluation.

Quelques précisions :

- ***la copie devra présenter une en-tête d'au moins une demi-page ainsi qu'une marge suffisante,***
- ***toutes les pages de la copie devront être numérotées et rangées dans l'ordre de lecture,***
- ***les résultats finaux doivent être clairement mis en évidence (soulignés ou encadrés),***
- ***les questions d'un même exercice doivent être présentées dans l'ordre du sujet.***

L'usage de tout matériel électronique est interdit. Aucun document n'est autorisé. Le sujet est à rendre avec la copie.

"Je crois beaucoup en la chance, et je constate que plus je travaille, plus elle me sourit."
Thomas Jefferson

n est un entier naturel non nul, X et Y sont des variables aléatoires sur un certain univers Ω, à valeurs dans $[\![0;n]\!]$, $\omega \in \Omega$.

1. $\mathbb{P}(X)$: HORREUR
2. $[X=0]$: évènement (ensemble d'issues)
3. $X(\Omega)$: ensemble de réels
4. $X(\omega)$: réel
5. $[X=1]+[X=2]$: HORREUR
6. X^2 : variable aléatoire
7. $\bigcap_{k=0}^{n}[X=k]$: évènement
8. $\mathbb{P}([X=0]) \cap \mathbb{P}([X=1])$: HORREUR
9. $\mathbb{P}_{[X=0]}\big([X=1]\cup[X=2]\big)$: réel
10. $\mathbb{P}([X+Y=2])$: réel
11. $\mathbb{E}([X=1])$: HORREUR
12. $\mathbb{E}(X^2)$: réel

Rappels :

- Une probabilité $\mathbb{P}$ est une application de $\mathcal{P}(\Omega)$ dans $[0;1]$ vérifiant deux conditions. On ne calcule donc la probabilité que d'un **évènement**. La probabilité d'un évènement est un réel de $[0;1]$.
- Une variable aléatoire X est une application de Ω dans $\mathbb{R}$: à chaque issue, elle associe un réel. La notation $[X=1]$ désigne l'ensemble des issues dont l'image par X est égale à 1 (en écriture mathématique : $[X=1]=\{\omega \in \Omega \ / \ X(\Omega)=1\}$) : c'est un évènement. Les opérations habituelles sur les ensembles ont donc du sens sur les évènements : union, intersection, contraire.

Exercice 1

On considère la suite $(h_n)_{n\in\mathbb{N}^*}$ définie par : $\forall n \in \mathbb{N}^*,\ h_n = \sum_{k=1}^{n} \frac{1}{k}$.

1. Écrire une fonction `Python` telle que, pour tout $n \in \mathbb{N}^*$, l'exécution de `suite_h(n)` renvoie la valeur de h_n.
2. **Étude de la suite $(h_n)_{n\in\mathbb{N}^*}$.**
   - 2.a. Déterminer le sens de variation de la suite $(h_n)_{n\in\mathbb{N}^*}$.
   - 2.b. Démontrer que pour tout $x \in]-1;+\infty[$, $\ln(1+x) \leq x$.
   - 2.c. Justifier que pour tout $k \in \mathbb{N}^*$, $\ln\left(1+\frac{1}{k}\right) \leq \frac{1}{k}$. En déduire : $\forall n \in \mathbb{N}^*,\ h_n \geq \ln(n+1)$. Déterminer alors la limite de la suite $(h_n)_{n\in\mathbb{N}^*}$.
3. On considère les suites $(u_n)_{n\in\mathbb{N}^*}$ et $(v_n)_{n\in\mathbb{N}^*}$ définies sur par :

$$\forall n \in \mathbb{N}^*,\ u_n = h_n - \ln(n)\ ;\ v_n = h_n - \ln(n+1)$$

   - 3.a. A l'aide du résultat de la question 2.b., établir :

$$\forall n \in \mathbb{N}^*,\ \ln\left(\frac{n+2}{n+1}\right) \leq \frac{1}{n+1} \leq \ln\left(\frac{n+1}{n}\right)$$

   - 3.b. Montrer que les suites $(u_n)_{n\in\mathbb{N}^*}$ et $(v_n)_{n\in\mathbb{N}^*}$ convergent toutes deux vers la même limite, notée γ.
   - 3.c. Établir :

$$\lim_{n\to+\infty} \frac{h_n}{\ln(n)} = 1$$

   - 3.d. Justifier que pour tout $n \in \mathbb{N}^*$, $v_n \leq \gamma \leq u_n$.
   - 3.e. Écrire une fonction `Python` prenant en argument d'entrée un réel strictement positif p et renvoyant en sortie un encadrement de γ d'amplitude inférieure ou égale à p (cette fonction pourra utiliser la fonction de la question 1.).
4. On pose, pour tout $n \in \mathbb{N}^*$, $S_n = \sum_{k=1}^{2n} \frac{(-1)^{k-1}}{k}$.
   - 4.a. Montrer par récurrence : $\forall n \in \mathbb{N}^*,\ S_n = h_{2n} - h_n$.
   - 4.b. En déduire : $\forall n \in \mathbb{N}^*,\ S_n = u_{2n} - u_n + \ln(2)$.
   - 4.c. Conclure que la suite $(S_n)_{n\in\mathbb{N}^*}$ converge vers $\ln(2)$.

★★★★★★★

Exercice 2

On considère deux urnes :

- une urne rouge, $\mathcal{U}_R$, composée de deux balles rouges et deux balles bleues
- une urne bleue, $\mathcal{U}_B$, composée d'une balle rouge et trois balles bleues

Le but de l'exercice est d'étudier deux jeux de tirage dans ces urnes, avec ou sans remise.
Pour $n \in \mathbb{N}^*$, on notera R_n l'évènement "on tire une balle rouge au n-ième tirage" et B_n l'évènement : "on tire une balle bleue au n-ième tirage". Le premier tirage s'effectue toujours dans l'urne rouge ; puis le tirage numéro n s'effectuera dans l'urne de la couleur de la balle obtenue au tirage numéro $n-1$. On suppose qu'il y a équiprobabilité du choix des différentes balles.

Les parties A et B sont indépendantes entre elles.

Partie A

Dans cette partie, on effectue une **succession de tirages avec remise** selon le protocole décrit au début de l'exercice.
On notera également, pour $n \in \mathbb{N}^*$, $r_n = \mathbb{P}(R_n)$ et $b_n = \mathbb{P}(B_n)$.

1. Soit $n \in \mathbb{N}^*$. Justifier que $\mathbb{P}(R_n) \neq 0$ puis $\mathbb{P}_{R_n}(R_{n+1}) = \frac{1}{2}$.
2. Démontrer que pour tout $n \in \mathbb{N}^*$: $r_{n+1} = \frac{1}{4} r_n + \frac{1}{4}$.
3. En déduire le terme général de $(r_n)_{n\in\mathbb{N}^*}$ puis celui de $(b_n)_{n\in\mathbb{N}^*}$.

Partie B

Dans cette partie, on effectue **trois tirages successifs sans remise** selon le protocole décrit précédemment.

1. Justifier que $\mathbb{P}_{R_1}(R_2) = \frac{1}{3}$. Déterminer également $\mathbb{P}_{B_1}(R_2)$.
2. Calculer $\mathbb{P}(R_2)$.
3. Calculer $\mathbb{P}_{R_2}(R_1)$.
4. Que dire de l'évènement $R_1 \cap R_2 \cap R_3$?
5. On note X la variable aléatoire égale au nombre de balles rouges obtenues durant ces trois tirages.
   - 5.a. Donner $X(\Omega)$. *Justifier.*
   - 5.b. Déterminer $\mathbb{P}_{B_1 \cap B_2}(B_3)$. En déduire $\mathbb{P}([X = 0])$.
   - 5.c. Calculer également $\mathbb{P}([X = 1])$.
   - 5.d. Vérifier que $\mathbb{P}([X = 2]) = \frac{1}{3}$.
   - 5.e. Calculer $\mathbb{E}(X)$ et $\mathbb{V}(X)$.
6. On note Y la variable aléatoire égale au nombre de balles bleues obtenues durant ces trois tirages.
   - 6.a. Exprimer Y en fonction de X.
   - 6.b. En déduire $\mathbb{P}([Y = 1])$ ainsi que $\mathbb{E}(Y)$ et $\mathbb{V}(Y)$.

★★★★★★★

Exercice 3

Soit n un entier naturel supérieur ou égal à 3.
Une urne contient une balle noire non numérotée et $n-1$ balles blanches, dont $n-2$ portent le numéro 0 et une porte le numéro 1. On extrait ces balles au hasard, une à une, sans remise, jusqu'à l'apparition de la balle noire.
Pour tout $i \in [\![1, n-1]\!]$, on note B_i l'événement : "le i-ème tirage donne une balle blanche", on pose $N_i = \overline{B_i}$ et on note X_n la variable aléatoire égale au rang d'apparition de la balle noire.

1. Donner l'ensemble $X_n(\Omega)$ des valeurs que peut prendre la variable X_n.
2. - 2.a. Pour tout $i \in [\![2; n-1]\!]$, déterminer $\mathbb{P}_{B_1 \cap \cdots \cap B_{i-1}}(B_i)$.
   - 2.b. Établir alors :
   $$\forall k \in X_n(\Omega),\ \mathbb{P}([X_n = k]) = \frac{1}{n}$$
   - 2.c. Calculer l'espérance et la variance de X_n, notées respectivement $\mathbb{E}(X_n)$ et $\mathbb{V}(X_n)$.
3. On note Y la variable aléatoire qui vaut 1 si la balle numérotée 1 a été piochée lors de l'expérience précédente, et qui vaut 0 sinon.
   - 3.a. Pour tout $k \in X_n(\Omega)$, montrer que :
   $$\mathbb{P}\Big([X_n = k] \cap [Y = 0]\Big) = \frac{n-k}{n(n-1)}$$
   - 3.b. En déduire, grâce à la formule des probabilités totales, la valeur de $\mathbb{P}([Y = 0])$.
   - 3.c. Déduire alors la loi de Y.
4. - 4.a. Recopier et compléter le script `Python` suivant de sorte que l'exécution de `simul_X(n)` simule une réalisation de l'expérience décrite ci-dessus, où n est le nombre total de balles, et renvoie la valeur de X_n associée. *On admettra que la balle noire est codée tout au long de ce script par le nombre 1.*

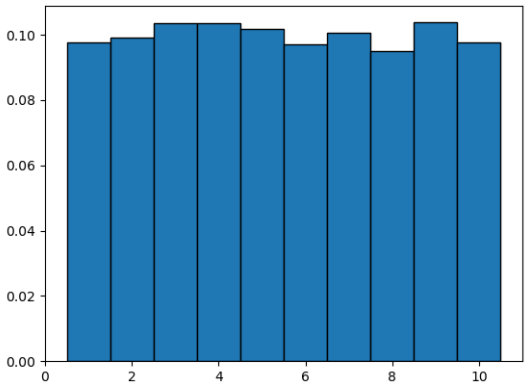
```
import numpy.random as rd
import matplotlib.pyplot as plt

def simul_X(n):
    N=n
    u=rd.randint(1,N+1)
    X=1
    while u!=1:
        N=...
        u=...
        X=...
    return X
```

   - 4.b. En utilisant la fonction créée à la question précédente, écrire une fonction `Python` telle que l'exécution de `esp_var_X(n)` renvoie une valeur approchée de $\mathbb{E}(X_n)$ et une de $\mathbb{V}(X_n)$.
   - 4.c. Recopier et compléter le programme suivant afin que son exécution affiche l'histogramme obtenu à partir de 10000 réalisations de la variable aléatoire X_n, où n est saisi par l'utilisateur.

```
n=int(input("n=?"))
Labs=
LX=
plt.hist(LX,Labs,edgecolor='k',density=True)
plt.show()
```

4.d. On a exécuté le programme de la question précédente en saisissant $n = 10$ et on a obtenu le graphique qui suit. Expliquer en quoi le graphique est cohérent avec la loi de X_n obtenue en question 2.b.



4.e. Écrire une fonction de sorte que l'exécution de `simul_XY(n)` simule une réalisation de l'expérience décrite ci-dessus, où n est le nombre total de balles, et renvoie les valeurs de X_n et Y associées.

★★★★★★★

Exercice 4

Dans toute la suite de l'exercice, on note $(F_n)_{n\in\mathbb{N}}$ la suite de Fibonacci définie par :

$$\begin{cases} F_0 = 0 \ ; \ F_1 = 1 \\ \forall n \in \mathbb{N},\ F_{n+2} = F_{n+1} + F_n \end{cases}$$

Partie A. Étude de la suite de Fibonacci.

1. Établir : $\forall n \in [\![2;+\infty[\![,\ 0 < F_n < F_{n+1}$.
2. Déterminer, pour tout $n \in \mathbb{N}$, l'expression de F_n en fonction de n.
3. Montrer alors que la suite (F_n) diverge vers $+\infty$.

Partie B. Deux programmes Python.

4. Écrire une fonction `Python` telle que, pour tout $n \in \mathbb{N}$, l'exécution de `fibo(n)` renvoie la valeur de F_n.
Cette fonction ne doit pas être récursive.
Si on exécute le script `Python` suivant

```
L=[fibo(k) for k in range(20)]
print (L)
```

on doit obtenir :

```
[0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987, 1597, 2584, 4181]
```

5. Écrire une fonction `Python` nommée `recherche` prenant en arguments d'entrée :
   - un réel x,
   - une liste `L` de réel triés dans l'ordre croissant, dont le premier élément est inférieur ou égal à x et le dernier strictement supérieur à x,

   et renvoyant en sortie le plus grand élément de `L` qui soit inférieur ou égal à x.

Partie C. Le théorème de Zeckendorf.

Dans cette partie, on s'intéresse au théorème suivant appelé **théorème de Zeckendorf** :

> Pour tout entier naturel non nul n, il existe un unique entier non nul k et un unique k-uplet d'entiers $(c_1;\ldots;c_k)$ tels que :
> - $c_1 \geq 2$ et $\forall i \in [\![1;k-1]\!],\ c_i + 1 < c_{i+1}$
> - $n = \sum_{i=1}^{k} F_{c_i}$, où (F_n) est la suite de Fibonacci précédemment introduite.

Une telle décomposition d'un entier naturel non nul n s'appelle **décomposition de Zeckendorf de n**.
Exemples :

- Pour $n = 4$: on remarque que $4 = 1 + 3 = F_2 + F_4$. Donc $k = 2$ et $(c_1 ; c_2) = (2; 4)$.
- Pour $n = 17$: on remarque que $17 = 1 + 3 + 13 = F_2 + F_4 + F_7$. Donc $k = 3$ et $(c_1 ; c_2 ; c_3) = (2; 5; 7)$.

6. *On rappelle que la liste des premiers termes de la suite de Fibonacci a été donnée en question 12.*

6.a. En remarquant que $6 = 1+2+3 = F_2+F_3+F_4$ et que $6 = 1+5 = F_2+F_5$, donner la décomposition de Zeckendorf de 6. Justifier.

6.b. Donner, sans justifier, les décompositions de Zeckendorf de 88 et 233.

7. *Dans cette question, nous allons démontrer l'existence d'une telle décomposition.* Procédons par récurrence forte. Le résultat est bien évidemment valable pour 1. Soit $n \in \mathbb{N}^*$. Supposons que "tout entier $m \in [\![1; n]\!]$ admet une décomposition de Zeckendorf" et montrons que "$n+1$ admet une décomposition de Zeckendorf".

7.a. Justifier l'existence d'un entier J supérieur ou égal à 4 tel que : $\forall i \geq J,\ F_i \geq n+2$.

7.b. Notons $E_n = \{i \in \mathbb{N}^* \ / \ F_i \leq n+1\}$. Montrer que $3 \in E_n$ et que E_n contient au plus $J-1$ éléments. *On note alors j le plus grand élément de E_n.*

7.c. Justifier que $j \geq 3$ et que $F_j \leq n+1 < F_{j+1}$.

7.d. Posons $m = n+1-F_j$. Montrer que $m < F_{j-1}$ puis conclure sur la décomposition de Zeckendorf de $n+1$.

8. **8.a.** Expliquer ce que renvoie l'exécution de la fonction `Python` suivante.

```
def liste_fibo(n):
    L=[0,1]
    while L[-1]<=n:
        L.append(L[-2]+L[-1])
    return L
```

Pourquoi préférer le programme précédent au programme suivant (où `fibo` est la fonction créée à la question 12.)?

```
def liste_fibo_bis(n):
    i=0
    L=[fibo(i)]
    while L[-1]<=n:
        i=i+1
        L.append(fibo(i))
    return L
```

8.b. En utilisant les fonctions `recherche` et `liste_fibo` des questions précédentes, écrire un algorithme glouton de sorte que, pour tout $n \in \mathbb{N}^*$, l'exécution de la commande `Zeckendorf(n)` renvoie la décomposition de Zeckendorf de n.