TD8

SEMAINE DE RÉVISIONS

1. Variables aléatoires discrètes.

1.1. Fonction génératrice d'une variable aléatoire.

Exercice 1

Soit X une variable aléatoire discrète. On définit la *fonction génératrice* G associée à X par

$$G(t) = \sum_{k \in X(\Omega)} \mathbb{P}([X = k])\, t^k,$$

pour tout réel t tel que la série converge.

Déterminer la fonction génératrice et leur domaine de définition dans les cas suivants :

1. X suit une loi de Bernoulli de paramètre $p \in]0, 1[$.
2. X suit une loi uniforme sur $[\![1, n]\!]$.
3. X suit une loi binomiale de paramètres $n \in \mathbb{N}^*$ et $p \in]0, 1[$.
4. X suit une loi géométrique de paramètre $p \in]0, 1[$.
5. X suit une loi de Poisson de paramètre $\lambda \in \mathbb{R}_+^*$.

Exercice 2

Soit a, b et N trois entiers supérieurs ou égaux à 2 tels que $N = a + b$. On considère une urne contenant initialement a boules rouges et b boules vertes. On effectue des tirages successifs dans cette urne, au hasard et avec remise en procédant de la façon suivante :

- Si la boule tirée est rouge, elle est remise dans l'urne avant de procéder au tirage suivant.
- Si la boule tirée est verte, elle n'est pas remise dans l'urne mais remplacée par une boule rouge, et on procède au tirage suivant.

On note Y la variable aléatoire égale au nombre de tirages nécessaires à l'obtention de la première boule rouge.

1. Déterminer la loi de Y. Montrer alors que pour tout $k \in [\![1, b]\!]$,
$$\mathbb{P}([Y = k]) = \frac{b!}{N^b}\left(\frac{N^{b-k+1}}{(b-k+1)!} - \frac{N^{b-k}}{(b-k)!}\right).$$
2. Soit G la fonction définie sur $\mathbb{R}$ par
$$G(x) = \sum_{k \in Y(\Omega)} \mathbb{P}([Y = k])\, x^k.$$
On dit que G est la fonction génératrice de la variable Y.

a. Justifier l'égalité $G(x) = \mathbb{E}(x^Y)$.

b. Quelle est la valeur de $G(1)$?

c. Exprimer $\mathbb{E}(Y)$ en fonction de $G'(1)$.

d. Exprimer la variance de Y en fonction de $G'(1)$ et de $G''(1)$.

3. Montrer que pour tout $x \in \mathbb{R}$,

$$G(x) = 1 + \frac{b!}{N^b}(x-1)\sum_{k=0}^{b}\frac{N^{b-k}}{(b-k)!}x^k.$$

4. En déduire l'espérance de Y.

On laissera le résultat sous forme d'une somme.

5. De même calculer la variance de Y à l'aide de la fonction génératrice G.

On laissera le résultat sous forme d'une somme.

Exercice 3 **ECRICOME** 2008.

Un joueur lance successivement n boules au hasard dans N cases numérotées de 1 à N (avec $N \geqslant 2$), chaque boule ayant une probabilité $1/N$ de tomber dans chacune des N cases (et les lancers de boules sont indépendants les uns des autres). On cherche à étudier la variable aléatoire T_n, égale au nombre de cases non vides après n lancers.

1. Déterminer en fonction de n et de N les valeurs prises par T_n.

2. Donner les lois de T_1 et T_2.

3. Déterminer, lorsque $n \geqslant 2$, les probabilités $\mathbb{P}([T_n = 1])$, $\mathbb{P}([T_n = 2])$ et $\mathbb{P}([T_n = n])$ (en distinguant suivant que $n \leqslant N$ ou $n > N$).

4. À l'aide de la formule des probabilités totales, prouver que si $1 \leqslant k \leqslant n$,

$$\mathbb{P}([T_{n+1} = k]) = \frac{k}{N}\mathbb{P}([T_n = k]) + \frac{N-k+1}{N}\mathbb{P}([T_n = k-1]).$$

5. On considère dans les questions qui suivent le polynôme

$$G_n(x) = \sum_{k=1}^{n}\mathbb{P}([T_n = k])\,x^k.$$

Quelle est la valeur de $G_n(1)$?

6. En utilisant la relation démontrée à la question **4**, montrer que

$$G_{n+1}(x) = \frac{1}{N}(x - x^2)G_n'(x) + xG_n(x).$$

7. Dériver l'expression précédente et en déduire que

$$\mathbb{E}(T_{n+1}) = \left(1 - \frac{1}{N}\right)\mathbb{E}(T_n) + 1.$$

8. Prouver enfin que

$$\mathbb{E}(T_n) = N\left(1 - \left(1 - \frac{1}{N}\right)^n\right),$$

et déterminer la limite de $\mathbb{E}(T_n)$ quand n tend vers $+\infty$.

1.2. **Autour de la formule des probabilités composées.**

Exercice 4

Une pièce de monnaie est déséquilibrée de sorte que la probabilité d'apparition de Pile est égale à $\frac{1}{3}$. On effectue avec cette pièce une suite de lancers indépendants et on considère, pour tout $n \geqslant 2$, l'événement

A_n : "La séquence Pile-Face apparaît pour la première fois au $n-1$-ième et n-ième lancers".

Pour tout $n \geqslant 2$, on note a_n la probabilité de l'événement A_n.

1. Calculer a_2, a_3 et a_4.
2. Montrer que pour tout $n \geqslant 2$, $a_n = \sum_{k=1}^{n-1} \frac{2^k}{3^n}$. Calculer a_n pour tout $n \geqslant 2$.
3. Calculer la probabilité que la sequence Pile-Face n'apparaisse jamais.
4. On considère alors la variable aléatoire X égale au nombre de lancers nécessaires pour qu'apparaisse pour la première fois la séquence Pile-Face. Déterminer la loi de X et son espérance.
5. **a.** Écrire un programme Python qui simule la variable X.
   b. Comment obtenir, informatiquement, une valeur approchée de $\mathbb{E}(X)$?
6. **a.** Soit, pour tout $n \geqslant 2$, l'événement

   B_n : "La séquence Pile-Face apparaît pour la première fois au $n-1$-ième lancer et au n-ième lancers et il n'y a pas eu avant de séquence Face-Pile.

   Pour tout $n \geqslant 2$, on note b_n la probabilité de l'événement B_n. Calculer b_n pour tout $n \geqslant 2$.
   b. En déduire la probabilité pour que la première séquence Pile-Face apparaisse avant la première séquence Face-Pile.

1.3. **Autour de la formule des probabilités totales.**

Exercice 5

On considère deux pièces de monnaie notées A_1 et A_2.

- Lorsqu'on lance la pièce A_1, la probabilité d'obtenir face est p_1, celle d'obtenir pile est $q_1 = 1 - p_1$.
- Lorsqu'on lance la pièce A_2, la probabilité d'obtenir face est p_2, celle d'obtenir pile est $q_2 = 1 - p_2$.

On effectue une suite de parties de la façon suivante :

- À la première partie on choisit une pièce au hasard avec probabilité 1/2 et on joue avec cette pièce :
  - Si le résultat est face, on joue la deuxième partie avec A_1.
  - Si le résultat est pile, on joue la deuxième partie avec A_2.
- Ensuite, pour tout entier $n \geqslant 1$,
  - Si on a obtenu face à la n-ième partie, on joue la $(n+1)$-ième avec la pièce A_1.
  - Si on a obtenu pile à la n-ième partie, on joue la $(n+1)$-ième avec la pièce A_2.

1. Pour tout entier $n \geqslant 1$, on note u_n la probabilité d'avoir face à la n-ième partie.
   a. Exprimer u_1 et u_2 en fonction de p_1 et p_2.
   b. Montrer que pour tout entier $n \geqslant 1$, $u_{n+1} = (p_1 - p_2)u_n + p_2$.
   c. Montrer que la suite $(u_n)_{n\in\mathbb{N}^*}$ tend, quand n tend vers l'infini, vers une limite u que l'on calculera. Dans quels cas a-t-on $u = \frac{1}{2}$?
2. Pour tout entier $n \geqslant 1$, on note X_n la variable aléatoire, associée à la n-ième partie, qui prend la valeur 1 si le résultat de la n-ième partie est face et la valeur 0 si le résultat est pile.
   a. Déterminer les lois de probabilité des variables X_1 et X_2 et calculer leurs espérances.
   b. Les variables aléatoires X_1 et X_2 sont-elles indépendantes ?

La suite de cet exercice est dans le sujet de HEC 1982 Maths III.

2. Révisions Python.

2.1. **Compléments sur les simulations de variables aléatoires.** On commence, pour toute la suite de ce TP à importer les bibliothèques et libraires nécessaires à chaque situation :

```
import numpy as np
import numpy.random as rd
```

2.1.1. *Un exemple de n-lancer d'une pièce de monnaie.* On désigne par n un entier naturel supérieur ou égal à 2. On lance n fois une pièce équilibrée (c'est-à-dire qui donne "pile" avec probabilité $\frac{1}{2}$ et "face" avec probabilité $\frac{1}{2}$), les lancers étant supposés indépendants. On note Z la variable aléatoire égale à 0 si on obtient aucun "pile" pendant ces n lancers et qui, dans le cas contraire, prend la valeur du premier "pile".

1. Rappeler comment simuler l'événement "la pièce tombe sur pile" à l'aide de la fonction rd.rand().
2. Dans la fonction Python suivante, on simule l'expérience en procédant à une suite constituée *a priori* de ces n lancers. À chaque expérience, si on tombe sur "pile", alors on arrête l'expérience et on renvoie le numéro du lancer, sinon on continue les lancers. Si à la fin des n lancers, on n'est jamais tombé sur "pile", alors on renvoie 0. Compléter cette fonction

```
def sumul_Z(n) :
  for i in range(1,n+1) :
    if ........... :
      return
  return
```

3. Dans la fonction Python suivante, on simule l'expérience en procédant à une suite constituée *a priori* d'une infinité de lancers. On effectue cette suite infinie tant qu'on obtient "face" et on compte alors le nombre de lancers effectuées pour obtenir le premier "pile". Si ce nombre est inférieur ou égal à n, Z prend cette valeur, sinon elle prend la valeur 0. Compléter cette fonction.

```
def sumul2_Z(n) :
  Z = 1
  while ............... :
    Z = Z + 1
  if .................. :
    return ..........
  else :
    return
```

4. Calculer, pour tout $k \in [\![0, n]\!], \mathbb{P}([Z = k])$. On traitera à part le cas $k = 0$.

2.1.2. *Un exemple de suite de tirages dans une urne.* Une urne contient initialement deux boules rouges et une boule bleue indiscernables au touché. On appelle pioche au hasard une boule dans l'urne puis

- Si la boule piochée est bleue, on la remet dans l'urne.
- Si la boule piochée est rouge, on ne la remet pas dans l'urne et on ajoute une boule bleue.

Ainsi le nombre de boules ne change pas à l'issue d'un tirage.

Pour tout entier naturel n, on note Y_n la variable aléatoire égale au nombre de boules rouges présentes dans l'urne à l'issue du n-ième tirage.

1. Soit $n \in \mathbb{N}^*$. Donner $Y_n(\Omega)$.
2. Compléter la fonction suivante, pour que, prenant en argument un entier $n \geqslant 1$, elle renvoie une simulation de la variable Y_n.

```
def sumul_Y(n) :
  r = 2
  for k in range(n) :
    if ............... :
      if ............... :
        r = r-1
  return ...........
```

3. On introduit pour tout $n \geqslant 1$, la variable aléatoire indicatrice de l'événement $[Y_n = 0]$, que l'on note T_n. Autrement dit,

$$T_n = \begin{cases} 1 & \text{si l'événement } [Y_n = 0] \text{ est réalisé} \\ 0 & \text{sinon} \end{cases}.$$

Quelle est la loi de la variable T_n ?

4. Écrire une fonction simul_T qui

- prend en argument un entier $n \geqslant 1$.
- renvoie une simulation de la variable T_n.

2.1.3. *Une loi usuelle ?* Soit $N \geqslant 3$ un entier. On considère une urne qui contient $N-1$ boules blanches et une seule boule noire. On effectue des tirages sans remise jusqu'à l'obtention de la boule noire et on note X la variable aléatoire égale au rang du tirage où on obtient la boule noire.

1. Donner $X(\Omega)$. La variable aléatoire est-elle finie ou infinie ?

2. Écrire une fonction simul_X qui

- prend en argument un entier $N \geqslant 3$.
- renvoie une simulation de X.

3. Recopier et exécuter les instructions ci-contre. Expliquer ce que fait ce script.

```
N = 5  # on fera varier la valeur de N
comptage_val_X = np.zeros(N)
for k in range(10000) :
  i = simul_X(N)
  comptage_val_X[i-1] += 1
freq_val_X = comptage_val_X / 10000
plt.bar(range(1,N+1), freq_val_X)
plt.show()
```

4. En interprétant le tracé obtenu à la question précédente, faire une conjecture sur la loi de X.

5. Démontrer la conjecture précédente.

2.1.4. *Simulation de variable aléatoire à l'aide de lois usuelles.* À la fête foraine, un jeu est proposé, dont la partie joué coûte 10 euros. Le jeu consiste à lancer une pièce de monnaie jusqu'à tomber sur "pile". On note alors n le rang d'apparition de ce premier "pile". Si $n \geqslant 2$, le joueur ne gagne rien, tandis que si $n \geqslant 3$, le joueur gagne n^2 euros. On note X la variable aléatoire égale au gain algébrique du joueur.

1. Donner $X(\Omega)$. La variable est-elle finie ou infinie ?

2. Écrire une fonction simul_X qui

- prend en argument un réel $p \in]0,1[$ tel que la pièce tombe sur "pile" avec probabilité p.
- renvoie une simulation de X.

3. Sans calculer la loi de la variable X, proposer une stratégie informatique qui permette de décider si le jeu est favorable au joueur (en fonction de p).

2.2. Simulation de couples de variables discrètes.

2.2.1. *ECRICOME 2022.* On dispose de trois urnes U_1, U_2 et U_3 et d'une infinité de jetons numérotés 1,2,3,4,...

On répartit un par un les jetons dans les urnes : pour chaque jeton, on choisit au hasard et avec équiprobabilité une des trois urnes dans laquelle on place le jeton. Le placement de chaque jeton est indépendant de tous les autres jetons et la capacité des urnes en nombre de jetons n'est pas limité.

Pour tout entier naturel n non nul, on note $X_n^{(1)}$ (resp. $X_n^{(2)}$, $X_n^{(3)}$) le nombre de jetons présent dans l'urne 1 (resp. l'urne 2, l'urne 3) après avoir réparti les n premiers jetons.

Soit Y le nombre de jetons placés lorsque, pour la première fois, deux urnes sont occupées par au moins un jeton.

Soit Z le nombre de jetons placés lorsque, pour la première fois, les trois urnes contiennent chacune au moins un jeton.

1. Compléter la fonction Python suivante pour qu'elle simule les n premiers placements des jetons et qu'elle renvoie un tableau numpy contenant le résultat des simulations de $X_n^{(1)}$, $X_n^{(2)}$ et $X_n^{(3)}$.

```
def remplissage_urne(n) :
  R = ..............
  for k in ......... :
    indice = ...........
    R[indice] += 1
  return R
```

2. On note a (resp. b et c) le nombre de jetons présents dans l'urne 1 (resp. 2 et 3) à un moment donné de l'expérience. Montrer que
   - Au moins une des urnes est vide si et seulement si $abc = 0$.
   - Au moins deux urnes sont vides si et seulement si $(a+b)(b+c)(c+a) = 0$.

3. Compléter la fonction Python suivante pour qu'elle simule l'expérience jusqu'à ce que les trois urnes contiennent au moins un jeton et qu'elle renvoie une liste $[Y, Z]$ contenant le résultat des simulations de Y et Z.

```
def simul_YZ() :
  R = ..........
  rang = 0
  liste = []
  while ..................... :
    rang = rang + 1
    indice = ..............
    R[indice] += 1
  ................
  while ................ :
    rang = rang + 1
    indice = ..................
    R[indice] += 1
  ......................
  return liste
```

4. Écrire un programme qui simule un grand nombre de fois le couple (X, Y), puis donne une estimation de l'espérance de X, de celle de Y et du coefficient de corrélation de (X, Y).

2.2.2. *Une expérience en deux étapes.* On lance une pièce de monnaie équilibrée une infinité de fois. On note N le rang d'apparition du premier "pile" obtenu. Si le premier "pile" a été obtenu au rang n, on lance ensuite n fois un dé équilibré à 6 faces. On note alors X le nombre de 6 obtenus et S la somme des résultats obtenus.

1. Écrire une fonction Python, simulNX() qui renvoie une liste $[N, X]$ contenant le résultat de la simulation de N et de X.

2. Écrire une fonction Python, simulNS() qui renvoie une liste $[N, S]$ contenant le résultat de la simulation de N et de S. On calculera S à l'aide d'une boucle for.

2.2.3. *L'embarras du choix parmi les urnes.* Soit $n \geqslant 2$ un entier fixé. On dispose de n urnes numérotées de 1 à n. Pour chaque $k \in [\![1, n]\!]$, l'urne k est composée de k boules numérotées de 1 à k. On choisit une urne au hasard (uniformément) puis au tire une boule uniformément au hasard dans cette urne. On note

- X_n la variable aléatoire égale au numéro de l'urne choisie.
- Y_n la variable aléatoire égale au numéro de la boule tirée

1. Écrire une fonction simulXY qui prend en argument un entier $n \geqslant 2$ et qui renvoie une liste [X,Y] contenant le résultat de la simulation de X_n et Y_n.
2. Quelle est la loi de Y_n ?

2.2.4. *Le savant fou.* Un savant fou s'ennuie dans sa tour isolée et dispose d'une pièce usée dont la probabilité de tomber sur "pile" est $p \in]0,1[$. Il décide de se lancer dans une expérience aléatoire dont le protocole est décrit ci-dessous :

- Il lance la pièce jusqu'à ce qu'elle tombe sur "pile". On note N la variable aléatoire égale au rang du premier "pile".
- Si la pièce est tombée sur "pile" pour la première fois au n-ième lancer, alors il remplit une urne de la manière suivante. Pour tout $k \in [\![1, n]\!]$,
  - Si k est pair, alors il lance un dé à 6 faces. Il place alors ensuite autant de jetons numérotés k que de le résultat indiqué sur le dé.
  - Si k est impair, alors il lance un dé à 8 faces. Il place ensuite dans l'urne autant de jetons numérotés k que le résultat indiqué sur le dé.
- Une fois l'urne remplie, il procède à n tirages successifs et sans remise dans cette urne, note les résultats obtenus et fait leur somme. On note S la variable aléatoire égale au résultat obtenu.

On souhaite simuler le couple (N, S). Pour cela, on rappelle que

- Si k est un entier, la commande k % 2 == 0 renvoie True si k est pair et False si k est impair.
- Si L est une liste possédant plus de n éléments, alors la commande rd.choice(L,n, replace = False) simule une succession de n tirages sans remise dans cette liste et renvoie une liste contenant les résultats successifs.

Écrire une fonction Python qui renvoie une liste [N,S] contenant le résultat de la simulation de N et S.

2.2.5. *Un test de dépistage.* Chaque jour, le nombre de personnes subissant un dépistage dans une certaine pharmacie est modélisé par une variable aléatoire X suivant une loi de Poisson de paramètre $m = 5$. Chaque test a une certaine probabilité $p = \frac{1}{10}$ d'être positif et les tests sont indépendants les uns des autres. On note N le nombre de tests positifs un jour donné.

1. Écrire une fonction Python nommée simulN() qui simule la variable aléatoire N.
2. Créer un échantillon de taille 1000 de cette variable et le comparer avec un histogramme, à un échantillon de même taille d'une loi de Poisson de paramètre $mp = \frac{1}{2}$. Que peut-on conjecturer ?
3. Démontrer cette conjecture (ou relire le calcul du cours où une autre version de ce calcul a été fait).

2.3. **Sélection d'activités.**

2.3.1. *Le paradoxe des anniversaires.* L'objectif de cette activité est de visualiser l'évolution de la probabilité qu'au moins deux personnes d'un groupe de n individus fêtent leur anniversaire le même jour, en fonction du nombre d'individus dans le groupe.

On suppose, pour simplifier, qu'une année n'est constituée que de 365 jours et que les naissances sont répartis uniformément tout au long de l'année. On s'intéresse à la date mais pas à l'année (exemple : 10 janvier !)

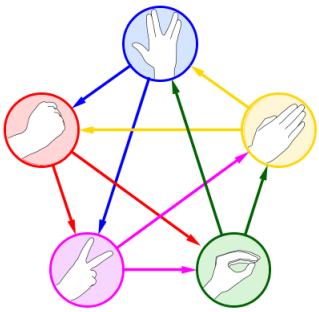
Déterminer graphiquement avec Python à partir de quelle valeur n, la probabilité qu'au moins deux personnes soient nées le même jour dépasse 1/2. Même question avec 99 %.

2.3.2. *Le code de César.* Jules César, général et stratège romain, a été (à ce qu'il parait) le premier militaire officiel à chiffrer ses messages. Sa méthode est assez simple : il décalait les lettres de 3 rangs dans l'alphabet.

On propose alors d'écrire une fonction de codage. Pour raffiner un peu le procédé élémentaire de César, la fonction devra prendre en argument le décalage (vers la droite ou vers la gauche), mais qui ne décale qu'une lettre sur deux (en commençant toujours par la deuxième lettre du mot). On aura en tête qu'en décalant 'z' de 1 vers la droite, on retombe sur 'a'.

Par exemple la commande codage(2,'amicalement') devra renvoyer aoieaneoept.

2.3.3. *Graphes Shifumi.* Inutile de rappeler les règles du *Shifumi* (ou Pierre-Feuille-Ciseaux). Ce jeu connaît aussi des variantes, comme celle (présentée plus bas) de la célèbre série télévisée *the Big Bang Theory*. Un



graphe de *Shifumi* est un graphe censé représenter les règles d'une confrontation. Les sommets sont les *figures* possibles. Il y a une arrête d'un sommet vers un autre s'il y a un vainqueur parmi les deux figures, et l'arrête est orientée pour pointer vers le perdant de la confrontation.

1. Représenter graphiquement, puis sous Python le graphe du *shifumi* classique.
2. Représenter le graphe d'un *shifumi* équilibré à 5 sommets (Pierre, Feuille Ciseaux, Lézard, Spock). Le représenter en *Python*.
3. Écrire une fonction Python appelée shifumi(G, nb_manches) qui permet de jouer un duel entre l'utilisateur et l'ordinateur (dont les figues sont choisies aléatoirement et de manière équiprobables) en nb_manches confrontations successives, et selon les règles représentées par le graphe G, et qui affiche le score final.
4. Soit n un entier impair. Écrire une fonction graphe_shifumi(n) qui renvoie un graphe de *shifumi* équilibré à n sommets.

2.3.4. *Le truel.* Trois gentleman, Mr. White, Mr. grey and Mr. Black se retrouvent opposés dans un *truel*, où chaque adversaire tire à son tour jusqu'à ce qu'il n'en reste qu'un.

Les trois hommes n'ont pas le même niveau de tire : Mr. Black fait mouche à chaque tire, Mr. grey touche son adversaire deux fois sur trois, tandis que Mr. White ne réussit son tire qu'une fois sur trois. Les hommes sont des gentlemen, ils laissent donc Mr. White tirer en premier, puis Mr. Grey et enfin Mr. Black.

Mr. Black visera toujours, tant que celui-ci sera vivant, Mr. Grey ; et Mr. Grey essaiera toujours lui-aussi de viser Mr. Black. Mr. White en revanche se demande s'il doit commencer par faire tomber Mr. Black, ou Mr. Grey, voire tirer en l'air et les laisser s'entretuer, donnant ainsi lieu à trois stratégies, numérotées 1, 2 et 3.

Comparer les trois stratégies avec Python, via simulation d'un grand nombre de truels pour les trois stratégies et émettre une conjecture sur celle la plus intéressante à suivre pour Mr. White. On pourra commencer par écrire une fonction qui simule les duels entre Mr. White et Mr. Grey avec dans chaque cas, un premier tireur différent.

2.3.5. *La ruine du joueur.* José se rend au casino *Les requins de la côte* avec s euros en poche ($s \in \mathbb{N}^*$) et l'envie de faire fortune. Après s'être vêtu de ses habits de lumière, il s'installe à une table où, à chaque partie, il gagne avec probabilité $p \in]0,1[$ 1 euro et perd avec probabilité $q = 1 - p$ un euro.

On note N la somme dont dispose le casino (on peut raisonnablement supposer que $s < N$). José décide qu'il arrêtera de jouer s'il devient ruiné (c'est-à-dire lorsque sa fortune tombe à 0) ou lorsque ce sera le cas pour le casino. On admet (même si on pourrait le montrer) que le jeu s'arrête à un moment presque sûrement.

1. Écrire une première fonction casino(s,N,p) qui à chaque appel renvoie le graphe de l'évolution de la fortune de José jusqu'à l'arrêt du processus.
2. On introduit la variable aléatoire T qui prend la valeur 1 si José est ruiné ou 0 si le casino est ruiné. Établir une conjecture sur la loi de T.

2.3.6. *Théorème de Zeckendorf.* On rappelle que la suite de Fibonacci $(F_n)_{n\in\mathbb{N}}$ est la suite de nombre réels définis par

$$F_0 = 0, \qquad F_1 = 1 \qquad \text{et} \qquad F_{n+2} = F_{n+1} + F_n.$$

Les premiers termes de la suite sont 0,1,1,2,3,5,8,13,21,34,55,...

Théorème : Théorème de Zeckendorf

Pour tout entier naturel $n \geqslant 1$, il existe un unique entier k et un unique k-uplet d'entiers $(c_1, c_2, \cdots, c_k)$ vérifiant

- $c_1 \geqslant 2$.
- $c_i + 1 < c_{i+1}$

tels que

$$n = \sum_{i=1}^{k} F_{c_i}.$$

Cette décomposition de n en somme de nombres de la suite de Fibonacci s'appelle la décomposition de Zeckendorf de n.

Par exemple, $17 = 13 + 3 + 1 = F_7 + F_4 + F_2$ donc $k = 3$ et $(c_1, c_2, c_3) = (2, 4, 7)$.

Le but de cette activité est d'écrire un programme qui renvoie cette décomposition pour un nombre n pris en argument.

1. Écrire une fonction `fibo(k)` qui renvoie le terme F_k de la suite $(F_n)_{n\in\mathbb{N}}$.
2. Écrire une fonction `tri(L)` qui renvoie la liste des termes de `L` triés par ordre croissant.
3. Écrire une fonction `recherche(x,L)` qui prend en argument un réel x et une liste L (déjà triée dans l'ordre croissant), de premier terme inférieur ou égal à x et de dernier terme strictement supérieur à x et qui renvoie le plus grand élément de la liste inférieur ou égal à x.
4. Écrire une fonction `Zeckendorf(n)` qui renvoie la décomposition de Zeckendorf de n.

   On pourra commencer par écrire la liste des nombres de Fibonacci inférieurs à n (ainsi que le premier strictement supérieur) et faire une boucle descendante sur cette liste.

3. Un sujet HEC/ESSEC.

Le but du problème est d'étudier le renouvellement d'un des composants d'un système complexe (une machine, un réseau de distribution d'énergie,...) formé d'un assemblage de différentes pièces susceptibles de tomber en panne. On s'intéresse donc à une de ces pièces susceptibles de se casser ou de tomber en panne et on se place dans la situation idéale où, dès que la pièce est défectueuse, elle est immédiatement remplacée. Dans une première partie, on étudie quelques propriétés fondamentales des variables aléatoires discrètes. Puis, dans une 2ème partie, on étudie la probabilité de devoir changer la pièce un certain jour donné. Enfin, dans une troisième partie, on cherche à estimer le temps de fonctionnement du système avec un certain nombre de pièces de rechange à disposition.

Dans tout le problème, on considère un espace probabilisé $(\Omega, \mathcal{A}, \mathbb{P})$. Pour toute variable aléatoire réelle X définie sur $(\Omega, \mathcal{A}, \mathbb{P})$, on note, sous réserve d'existence, $\mathbb{E}(X)$ son espérance et $\mathbb{V}(X)$ sa variance.

La 2ème partie peut être traitée en admettant si besoin les résultats de la première partie.

Première Partie. Dans cette première partie, on étudie les propriétés asymptotiques d'une variable aléatoire X à valeurs dans $\mathbb{N}^*$.

1. a. Montrer que pour tout entier naturel j non nul,

$$\mathbb{P}([X = j]) = \mathbb{P}([X > j - a]) - \mathbb{P}([X > j]).$$

b. Soit p un entier naturel non nul. Montrer que

$$\sum_{j=1}^{p} j\mathbb{P}([X = j]) = \sum_{j=0}^{p-1} \mathbb{P}([X > j]) - p\mathbb{P}([X > p]).$$

2. **a.** On suppose que X admet une espérance $\mathbb{E}(X) = \mu$.

(i) Justifier la convergence de la série de terme général $k\mathbb{P}([X = k])$.

(ii) Montrer que

$$\lim_{p\to+\infty} \sum_{k=p+1}^{+\infty} k\mathbb{P}([X = k]) = 0.$$

(iii) En déduire

$$\lim_{p\to+\infty} p\mathbb{P}([X > p]) = 0.$$

(iv) Montrer que la série de terme général $\mathbb{P}([X > j])$ converge.

(v) Montrer que $\mu = \sum_{j=0}^{+\infty} \mathbb{P}([X > j])$.

b. On suppose que $\sum_{j=0}^{+\infty} \mathbb{P}([X > j])$ converge.

(i) Déterminer le sens de variation de la suite $(v_p)_{p\in\mathbb{N}^*}$ définie par

$$v_p = \sum_{j=0}^{p-1} \mathbb{P}([X > j]).$$

(ii) Comparer $\sum_{j=1}^{p} j\mathbb{P}([X = j])$ et $\sum_{j=0}^{+\infty} \mathbb{P}([X > j])$.

(iii) En déduire que X admet une espérance.

c. Conclure des questions précédentes que X admet une espérance si et seulement si la série de terme général $\mathbb{P}([X > j])$ converge.

3. On suppose dans cette question qu'il existe un réel α strictement positif tel que, pour tout entier naturel j, on ait

$$\mathbb{P}([X > j]) = \frac{1}{(j+1)^\alpha} \quad (\star).$$

a. Légitimer que $(\star)$ définit bien une loi de probabilité d'une variable aléatoire à valeurs dans $\mathbb{N}^*$.

b. Montrer que X admet une espérance si et seulement si α est strictement supérieur à 1.

c. Montrer que pour tout entier naturel j non nul,

$$\mathbb{P}([X = j]) = \frac{1}{j^\alpha}\left(1 - \frac{1}{\left(1+\frac{1}{j}\right)^\alpha}\right).$$

d. (i) Étudier les variations de $f : x \longmapsto 1 - (1+x)^\alpha - \alpha x$ sur $[0, 1]$.

(ii) Montrer que pour tout entier naturel j non nul,

$$\mathbb{P}([X = j]) \leqslant \frac{\alpha}{j^{1+\alpha}}.$$

e. Montrer en utilisant le résultat de **3.c**, que

$$\lim_{j\to+\infty} j^{\alpha+1}\mathbb{P}([X = j]) = \alpha.$$

f. Montrer que X admet une variance si et seulement si $\alpha > 2$.

Deuxième partie : Étude de la probabilité de panne un jour donné. Dans cette deuxième partie, on suppose donnée une suite de variable aléatoires $(X_i)_{i\in\mathbb{N}^*}$ mutuellement indépendantes et de même loi à valeur dans $\mathbb{N}^*$.

Pour tout entier i non nul, X_i représente la durée de vie en jours du i-ème composant en fonctionnement.

Soit k un entier naturel non nul. On note $T_k = X_1 + \cdots + X_k$. La variable T_k représente donc le jour où le k-ème composant tombe en panne. On fixe un entier naturel n non nul représentant un jour donné et on considère l'événement A_n : "le composant en place le jour n tombe en panne", c'est-à-dire "il existe un entier naturel non nul k tel que $T_k = n$" et on se propose d'étudier $\mathbb{P}(A_n)$.

4. Pour tout entier naturel j, on note $p_j = \mathbb{P}([X_1 = j])$ et $u_j = \mathbb{P}(A_j)$. On suppose que pour tout entier naturel non nul j, on a $p_j \neq 0$. On pose de plus par convention $u_0 = 1$.

a. Montrer que $u_1 = p_1$.

b. (i) Montrer que $A_2 = [X_1 = 2] \cup ([X_1 = 1] \cap [X_2 = 1])$.

(ii) En déduire u_2 en fonction de p_1 et p_2.

c. Pour tout entier naturel i, on pose $\tilde{X}_i = X_{i+1}$.

(i) Montrer que les variables $\tilde{X}_i$ sont mutuellement indépendantes, indépendantes de X_1 et de même loi que X_1.

(ii) Soit k un entier naturel non nul strictement inférieur à n. Montrer que

$$A_n \cap [X_1 = k] = [X_1 = k] \cap \left(\bigcup_{j\geqslant 1} \left[\tilde{X}_1 + \tilde{X}_2 + \cdots + \tilde{X}_j = n - k \right] \right).$$

(iii) En déduire que pour tout entier naturel k non nul strictement inférieur à n

$$\mathbb{P}_{[X_1=k]}(A_n) = \mathbb{P}(A_{n-k}).$$

d. Montrer que

$$u_n = u_{n-1}p_1 + \cdots + u_0 p_n.$$

e. En Python, soit $P = [p_1, p_2, \cdots, p_n]$ le vecteur ligne tel que $P[j] = p_{j+1}$ pour tout j dans $[\![0, n-1]\!]$. Écrire un programme Python qui calcule u_n à partir de P.

5. Soit λ un réel appartenant à $]0, 1[$.

Dans cette question, on suppose que X suit la loi géométrique de paramètre λ. Pour tout entier naturel non nul, on a donc $\mathbb{P}([X_1 = j]) = \lambda(1-\lambda)^{j-1}$.

a. Calculer $\mathbb{P}([X_1 > k])$ pour tout entier naturel k non nul.

b. Calculer $\mathbb{P}_{[X_1>k]}([X_1 = k+1])$.

c. Montrer que pour tout entier naturel n non nul, $P(A_n) = \lambda$.

6. On suppose dans cette question que p_1 vérifie $0 < p_1 < 1$ et que $p_2 = 1 - p_1$.

Pour simplifier, on posera $p = p_1 = 1 - p_2$.

a. Que vaut p_i pour i supérieur à 3 ?

b. Soit la matrice $M = \begin{pmatrix} p & 1-p \\ 1 & 0 \end{pmatrix}$.

Montrer que pour tout entier naturel n supérieur ou égal à 2, $\begin{pmatrix} u_n \\ u_{n-1} \end{pmatrix} = M \begin{pmatrix} u_{n-1} \\ u_{n-2} \end{pmatrix}$.

c. (i) Diagonaliser la matrice M.

(ii) Montrer que

$$M^{n-1} = \frac{1}{2-p} \begin{pmatrix} 1 & 1-p \\ 1 & 1-p \end{pmatrix} + \frac{(p-1)^{n-1}}{2-p} \begin{pmatrix} 1-p & p-1 \\ -1 & 1 \end{pmatrix}.$$

d. (i) Exprimer u_n en fonction de p et de n.

(ii) Déterminer $\lim\limits_{n\to+\infty} u_n$.

Troisième partie : Étude de la durée de fonctionnement. Comme dans la partie précédente, on suppose donnée une suite de variables aléatoires $(X_i)_{i\in\mathbb{N}^*}$ indépendantes et de même loi, telle que, pour tout entier i non nul, X_i représente la durée de vie en jours du i-ème composant en fonctionnement.

Soit k un entier naturel non nul. On étudie dans cette partie la durée de fonctionnement prévisible du système si on a k composants à disposition (y compris celui installé au départ).

On notera toujours $T_k = X_1 + \cdots + X_k$.

On suppose dans cette partie qu'il existe un réel $\alpha > 1$ tel que, pour tout entier naturel j, on ait :

$$\mathbb{P}([X_1 > j]) = \frac{1}{(j+1)^\alpha}.$$

En particulier dans toute cette partie, X_1 admet une espérance et on notera $\mu = \mathbb{E}(X_1)$.

7. Que vaut $\mathbb{E}(T_k)$?

8. On suppose **dans cette question** que α est strictement supérieur à 2. La variable aléatoire admet donc une variance σ^2.

a. Calculer $\mathbb{V}(T_k)$.

b. Montrer que pour tout réel ε strictement positif,

$$\mathbb{P}([|T_k - k\mu| \geqslant k\varepsilon]) \leqslant \frac{\sigma^2}{k\varepsilon^2}.$$

c. Déduire que, pour tout réel strictement positif ε, on a :

$$\lim_{k\to+\infty} \mathbb{P}\left(\left[\frac{T_k}{k} \in]\mu-\varepsilon, \mu+\varepsilon[\right]\right) = 1.$$

9. On suppose maintenant uniquement que $\alpha > 1$ et donc que X_1 n'a pas nécessairement de variance d'où l'impossibilité d'appliquer la méthode précédente. On va mettre en oeuvre ce qu'on appelle une méthode de troncation.

On fixe un entier naturel m strictement positif. Pour tout entier naturel non nul i, on définit deux variables aléatoires $Y_i^{(m)}$ et $Z_i^{(m)}$ de la façon suivante :

$$Y_i^{(m)} = \begin{cases} X_i & \text{si } X_i \leqslant m \\ 0 & \text{sinon} \end{cases} \quad \text{et} \quad Z_i^{(m)} = \begin{cases} X_i & \text{si } X_i > m \\ 0 & \text{sinon} \end{cases}.$$

a. Montrer que $X_i = Y_i^{(m)} + Z_i^{(m)}$.

b. (i) En utilisant **3.d.**(ii), montrer que

$$\mathbb{E}\left(Z_1^{(m)}\right) \leqslant \sum_{i=m+1}^{+\infty} \frac{\alpha}{i^\alpha}.$$

(ii) Montrer que

$$\mathbb{E}\left(Z_1^{(m)}\right) \leqslant \int_m^{+\infty} \frac{\alpha}{x^\alpha} dx.$$

(iii) Calculer $\displaystyle\int_m^{+\infty} \frac{\alpha}{x^\alpha} dx$.

(iv) En déduire que $\lim_{m\to+\infty} \mathbb{E}\left(Z_1^{(m)}\right) = 0$.

(v) Montrer que $\lim_{m\to+\infty} \mathbb{E}\left(Y_1^{(m)}\right) = \mu$.

c. (i) Montrer que $\left(Y_1^{(m)}\right)^2 \leqslant mX_1$.

(ii) En déduire que $\mathbb{V}\left(Y_1^{(m)}\right) \leqslant m\mu$

d. Soit ε un réel strictement positif. Montrer qu'il existe un entier naturel m_0 non nul tel que pour tout entier naturel m supérieur ou égal à m_0,

$$\frac{\alpha}{\alpha-1}m^{1-\alpha} \leqslant .$$

Jusqu'à la fin du problème, m désigne un entier supérieur ou égal à m_0.

e. On note, pour tout entier naturel k non nul,

$$U_k^{(m)} = \sum_{i=1}^{k} Y_i^{(m)} \quad \text{et} \quad V_k^{(m)} = \sum_{i=1}^{k} Z_i^{(m)}.$$

Vérifier que

$$T_k = U_k^{(m)} + V_k^{(m)}.$$

f. (i) Montrer que $\mathbb{E}\left(V_k^{(m)}\right) \leqslant k\frac{\alpha}{\alpha-1}m^{1-\alpha}$.

(ii) En déduire que $\mathbb{P}\left(\left[V_k^{(m)} \geqslant k\varepsilon\right]\right) \leqslant \frac{\alpha}{\alpha-1}\frac{m^{1-\alpha}}{\varepsilon}$.

g. (i) Montrer que $\mathbb{E}\left(U_k^{(m)}\right) \geqslant k\mu - k\frac{\alpha}{\alpha-1}m^{1-\alpha}$.

(ii) En déduire que $\left|\mathbb{E}\left(U_k^{(m)}\right) - k\mu\right| \leqslant k\varepsilon$.

(iii) Montrer que

$$\mathbb{P}\left(\left[\left|U_k^{(m)} - k\mu\right| \geqslant 2k\varepsilon\right]\right) \leqslant \mathbb{P}\left(\left[\left|U_k^{(m)} - \mathbb{E}\left(U_k^{(m)}\right)\right| \geqslant k\varepsilon\right]\right).$$

(iv) Montrer que $\mathbb{V}\left(U_k^{(m)}\right) \leqslant km\mu$.

(v) En déduire que

$$\mathbb{P}\left(\left[\left|U_k^{(m)} - k\mu\right|\right]\right) \leqslant \frac{m\mu}{k\varepsilon^2}.$$

(i) Montrer que pour tout couple d'événements A et B dans $\mathcal{A}$, on a

$$\mathbb{P}(A \cap B) \geqslant \mathbb{P}(A) + \mathbb{P}(B) - 1.$$

(ii) En appliquant l'inégalité précédente aux événements

$$A = \left[V_k^{(m)} < k\varepsilon\right] \quad \text{et} \quad B = \left[U_k^{(m)} \in]k(\mu-2\varepsilon, k(\mu+2\varepsilon)[\right],$$

montrer que

$$\mathbb{P}\left([T_k \in]k(\mu-3\varepsilon, k(\mu+3\varepsilon)[]\right) \geqslant \mathbb{P}\left(\left[V_k^{(m)} < k\varepsilon\right]\right) + \mathbb{P}\left(\left[U_k^{(m)} \in]k(\mu-2\varepsilon, k(\mu+2\varepsilon)[\right]\right) - 1.$$

(iii) Déduire des questions précédentes que pour tout réel $\varepsilon > 0$ et pour tout entier $m \geqslant m_0$, on a, pour tout entier k non nul,

$$\mathbb{P}\left([T_k \in]k(\mu-3\varepsilon, k(\mu+3\varepsilon)[]\right) \geqslant 1 - \frac{\alpha m^{1-\alpha}}{\varepsilon(\alpha-1)} - \frac{m\mu}{k\varepsilon^2}.$$

(iv) Pour k assez grand, appliquer l'inégalité précédente à un entier $m_k \in \left[\sqrt{k}, 2\sqrt{k}\right]$ et conclure que

$$\lim_{k\to+\infty} \mathbb{P}\left(\left[\frac{T_k}{k} \in]\mu-3\varepsilon, \mu+3\varepsilon[\right]\right) = 1.$$