Exercice 1. Poule, Renard, Vipère

Dans une partie de "Poule, Renard, Vipère", chaque personne qui joue est soit une poule, soit un renard soit une vipère, et doit attraper ses proies sans se faire attraper par ses prédateurs. Les poules doivent attraper les vipères, les vipères doivent attraper les renards et les renards doivent attraper les poules. Lorsqu'un joueur se fait attraper, il passe dans le camp du joueur qui l'a attrapé (une poule devient un renard, un renard devient une vipère et une vipère devient une poule). Notre but est de modéliser ce jeu.

Le terrain est représenté par une grille de taille $n \times n$ sur laquelle les joueurs se déplacent vers le haut, vers le bas, vers la droite, vers la gauche ou en diagonale. Plusieurs joueurs peuvent se trouver sur la même case au même instant.

Initialement, chaque joueur est placé sur le terrain avec une probabilité uniforme et son camp est également choisi avec probabilité uniforme.

À chaque instant, la probabilité qu'un joueur en case (i, j) soit attrapé par un prédateur est égale au nombre de prédateurs sur la case (i, j) divisé par le nombre total de joueurs sur la case (i, j). Par exemple, si une case contient deux poules, un renard et une vipère, alors la probabilité pour chaque poule de devenir un renard est 0.25, la probabilité pour le renard de devenir une vipère est 0.25 et la probabilité pour la vipère de devenir une poule est 0.5.

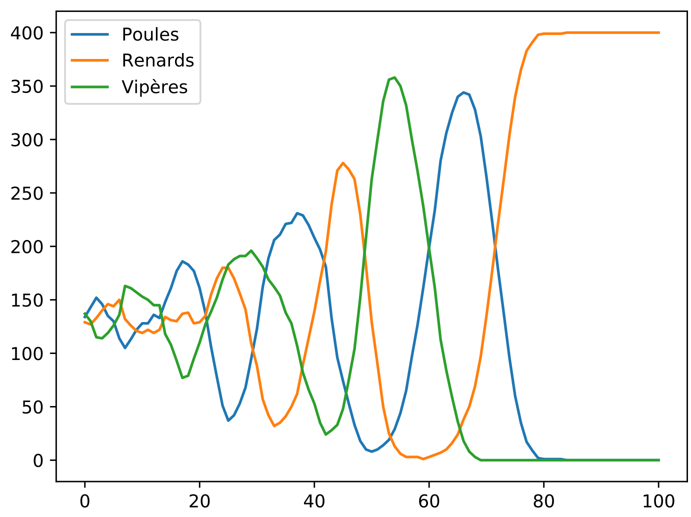
Une fois les changements de camps effectués, chaque joueur choisit au hasard uniformément une case voisine sur laquelle se déplacer. Une case est considérée comme voisine si c'est la case elle-même ou bien si un pas vertical, horizontal ou en diagonale suffit pour s'y rendre. Une case a donc au plus 9 cases voisines.

1. Écrire une fonction `jouer` qui prend en entrée n, le nombre de joueurs ainsi que le nombre d'étapes du jeu, et affiche dans une fenêtre graphique l'évolution du nombre de joueurs dans chaque camp en fonction de l'étape. Pour afficher la fenêtre graphique, on pourra utiliser le module `matplotlib.pyplot`. Le graphe ci-dessus correspond à $n = 10$, 400 joueurs et 100 étapes.

```python
import matplotlib.pyplot as plt

plt.figure()
# L1 contient le nombre de
# poules à chaque instant.
L1 = [...]
plt.plot(L1, label = 'Poules')
# L2 contient le nombre de
# renards à chaque instant.
L2 = [...]
plt.plot(L2, label = 'Renards')
# L3 contient le nombre de
# vipères à chaque instant.
L3 = [...]
plt.plot(L3, label = 'Vipères')

plt.legend()
plt.show()
```



   Pensez à organiser votre code avec des fonctions intermédiaires. Si besoin, on pourra lire les indications ci-dessous.

2. Refaire la question 1 en généralisant au cas où il y a c camps (on avait $c = 3$ jusqu'à présent). Pour tout $i \in [\![0, c-1]\!]$, le camp numéro i est la proie du camp numéro $(i + 1) \mod c$.

Indications (essayez de résoudre l'exercice sans lire ce qui suit). On aura besoin de connaître le nombre de poules de renards et de vipères présents sur chaque case. Ainsi, le terrain est représenté par une variable « `terrain: list[list[list[int]]]` » de taille $n \times n \times 3$ telle que `terrain[i][j][0]`, `terrain[i][j][1]` et `terrain[i][j][2]` sont le nombre de poules, renards et vipères sur la case (i, j). Dans la suite, on prendra comme exemple le cas où :

```
terrain0 = [[[1,0,0], [0,1,0], [1,1,1]],
            [[0,0,0], [2,0,0], [0,0,0]],
            [[0,0,0], [2,1,1], [0,0,0]]]
```

3. Écrire une fonction `terrain_initial` qui prend en entrée la taille du terrain n ainsi que le nombre de joueurs et renvoie un terrain de jeu tiré au hasard. On pourra utiliser la fonction `randint` du module `random`.
4. Écrire une fonction `nb_sur_terrain` qui prend en entrée la variable `terrain` et renvoie un triplet composé du nombre de poules, renards et vipères sur le terrain. Avec l'exemple ci-dessus, on obtient `(6,3,2)`.
5. Soit « `case: list[int]` » une liste de taille 3 représentant une case du terrain. Écrire une fonction `changement_camp_case` qui prend en entrée `case`, tire au hasard les joueurs qui changent de camp puis modifie `case` en conséquence. Votre fonction ne doit rien renvoyer.
6. Écrire une fonction `changement_camp` qui prend en entrée la variable `terrain`, et applique la fonction `changement_camp_case` à chaque case du terrain. Votre fonction ne doit rien renvoyer.
7. Écrire une fonction `voisins` qui prend en entrée n ainsi que deux entiers $i, j \in [\![0, n-1]\!]$ et renvoie la liste des coordonnées des cases voisines.
8. Écrire une fonction `deplacement` qui prend en entrée la variable `terrain` et modifie la position des joueurs entre deux étapes. Votre fonction ne doit rien renvoyer.

Exercice 2.

Commencer le sujet X/ENS 2022 (voir le lien sur la page du cours).

Exercices à rendre au plus tard le 08/03/2026 à 20h

Exercice 3. Grilles et carrés magiques

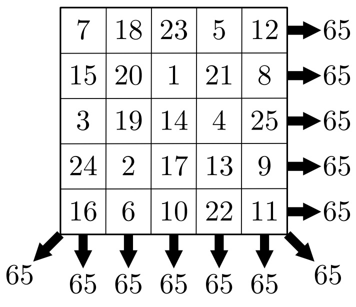


Figure 1

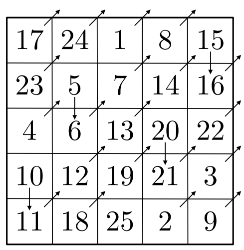


Figure 2

1	2
3	4

Figure 3

-2	-1
0	1

Figure 4

3	6
9	12

Figure 5

1	2	1	2	1	2
3	4	3	4	3	4
1	2	1	2	1	2
3	4	3	4	3	4
1	2	1	2	1	2
3	4	3	4	3	4

Figure 6

1	1	1	2	2	2
1	1	1	2	2	2
1	1	1	2	2	2
3	3	3	4	4	4
3	3	3	4	4	4
3	3	3	4	4	4

Figure 7

2	9	4
7	5	3
6	1	8

FIGURE 8

2	7	6
9	5	1
4	3	8

FIGURE 9

1	6	5
8	4	0
3	2	7

FIGURE 10

9	54	45
72	36	0
27	18	63

FIGURE 11

9	9	9	54	54	54	45	45	45
9	9	9	54	54	54	45	45	45
9	9	9	54	54	54	45	45	45
72	72	72	36	36	36	0	0	0
72	72	72	36	36	36	0	0	0
72	72	72	36	36	36	0	0	0
27	27	27	18	18	18	63	63	63
27	27	27	18	18	18	63	63	63
27	27	27	18	18	18	63	63	63

FIGURE 12

11	18	13	56	63	58	47	54	49
16	14	12	61	59	57	52	50	48
15	10	17	60	55	62	51	46	53
74	81	76	38	45	40	2	9	4
79	77	75	43	41	39	7	5	3
78	73	80	42	37	44	6	1	8
29	36	31	20	27	22	65	72	67
34	32	30	25	23	21	70	68	66
33	28	35	24	19	26	69	64	71

FIGURE 13

Définitions. Soit $n \in \mathbb{N}^*$. On appelle ***grille d'ordre*** n un tableau composé de n lignes et n colonnes. Chaque case de la grille est repérée par un couple (i, j) où i est l'indice de la ligne ($i = 0$ pour la ligne du haut, $i = n - 1$ pour la ligne du bas) et j est l'indice de la colonne ($j = 0$ pour la colonne de gauche, $j = n - 1$ pour la colonne de droite). Par exemple, la grille de la figure 1 est d'ordre $n = 5$, sa case de coordonnées $(1, 3)$ contient 21 et sa case de coordonnées $(4, 2)$ contient 10.

En Python, le type « `list[list[int]]` » sera noté « `grille` » et servira à représenter des grilles contenant des entiers. Ainsi, si « `G: grille` » correspond à une grille d'ordre n, alors `G` contient n sous-listes dont chacune est de taille n. Par exemple, la grille de la figure 1 correspond à :

```
G_fig1 = [[ 7, 18, 23,  5, 12], [15, 20,  1, 21,  8], [ 3, 19, 14,  4, 25],
          [24,  2, 17, 13,  9], [16,  6, 10, 22, 11]]
```

Produit de grilles. Soient $n \in \mathbb{N}^*$ et $m \in \mathbb{N}^*$ deux entiers, et G une grille d'ordre n.

- ⋆ On note $G - m$ la grille d'ordre n dans laquelle chaque case c a été remplacée par une case contenant $k - m$ où k est le contenu de c. Par exemple, si G est la figure 3 alors $G - 3$ est la figure 4 ; et si G est la figure 9 alors $G - 1$ est la figure 10.
- ⋆ On note $m \times G$ la grille d'ordre n dans laquelle chaque case c a été remplacée par une case contenant $m \times k$ où k est le contenu de c. Par exemple, si G est la figure 3 alors $3 \times G$ est la figure 5 ; et si G est la figure 10 alors $9 \times G$ est la figure 11.
- ⋆ On note $f_1(G, m)$ la grille d'ordre nm consistant à juxtaposer m^2 copies de G. Par exemple, si G est la figure 3 alors $f_1(G, 3)$ est la figure 6.
- ⋆ On note $f_2(G, m)$ la grille d'ordre nm consistant à remplacer chaque case c de G par m^2 cases contenant k où k est la valeur de c. Par exemple, si G est la figure 3 alors $f_2(G, 3)$ est la figure 7 ; et si G est la figure 11 alors $f_2(G, 3)$ est la figure 12.

Soit G_1 une grille d'ordre n_1 et G_2 une grille d'ordre n_2. Le “produit” de G_1 et G_2 est une grille G d'ordre $n_1 n_2$. Pour la construire, on définit :

$$H_1 = f_1(G_1, n_2) \qquad\qquad H_2 = f_2\left(n_1^2 \times (G_2 - 1), n_1\right)$$

La grille G est obtenue en sommant H_1 et H_2 case par case. Par exemple, si G_1 est la figure 8 et G_2 est la figure 9, alors $G_2 - 1$ est la figure 10, $n_1^2 \times (G_2 - 1)$ est la figure 11, H_2 est la figure 12 et G est la figure 13.

1. Écrire une fonction « `sous(G: grille, m: int) -> grille` » qui renvoie $G - m$.
2. À l'aide d'une compréhension de listes, écrire une fonction « `mult(G: grille, m: int) -> grille` » qui renvoie $m \times G$. Si vous ne vous souvenez plus de ce qu'est une compréhension, relisez le cours sur les listes.
3. Écrire une fonction « `f1(G: grille, m: int) -> grille` » qui renvoie $f_1(G, m)$.
4. Écrire une fonction « `f2(G: grille, m: int) -> grille` » qui renvoie $f_2(G, m)$.
5. Écrire une fonction « `produit(G1: grille, G2: grille) -> grille` » qui renvoie le produit de deux grilles.

Méthode siamoise (partie facultative). Un ***carré magique normal*** est une grille d'ordre n, où chaque entier de $[\![1; n^2]\!]$ apparaît une et une seule fois, et telle que les sommes des entiers sur chaque ligne, chaque colonne et chacune des deux diagonales soient égales à $\dfrac{n(n^2+1)}{2}$. Par exemple, la figure 1 est un carré magique normal d'ordre $n = 5$.

Soit $n \in \mathbb{N}^*$ un entier impair. La méthode siamoise permet de construire un carré magique normal d'ordre n. Pour cela, on part d'une grille initialement vide, puis on y place les nombres de 1 à n^2 de la manière suivante :

→ L'entier 1 est placé dans la case se trouvant au milieu de la première ligne.

→ Pour chaque $k \in [\![1, n^2 - 1]\!]$, après avoir placé k dans une case c, on s'intéresse à la case se trouvant au nord-est de c. Si cette case est vide, on y place l'entier $k + 1$, sinon on place $k + 1$ dans la case se trouvant au sud de c.

→ Lorsqu'on se déplace sur la grille, on considère que celle-ci est circulaire : les bords haut/bas sont reliés ainsi que les bords droit/gauche.

Par exemple, pour $n = 5$ on obtient la figure 2.

6. Écrire une fonction « `siamoise(n: int) -> grille` » qui renvoie le carré magique d'ordre `n` obtenu avec la méthode siamoise. Dans le cas où `n` est pair, votre fonction déclenchera une erreur.

Remarque. On peut montrer que le produit de deux carrés magiques est un carré magique.