Exercices - Probabilité avec Python

Nom :

EXERCICE 1. : JEU DU CHIFOUMI

a) Rappel des règles :

Les deux joueurs choisissent simultanément un des trois coups possibles en le symbolisant de la main :

Pierre

Feuille

Ciseau

De façon générale, la **pierre** bat les **ciseaux** (en les émoussant), les **ciseaux** battent la **feuille** (en la coupant), la **feuille** bat la **pierre** (en l'enveloppant). Ainsi chaque coup bat un autre coup, fait match nul contre le deuxième (son homologue) et est battu par le troisième.

b) Issues et Univers:

On suppose que 2 joueurs A et B s'affrontent sur une partie. Si le joueur A sort une **Pierre** et le joueur B sort un **Ciseau**, l'issue correspondante est notée **(pierre , ciseau). La première figure citée est pour le joueur A**.

L'univers Ω est l'ensemble qui contient toutes les issues possibles.

Question 1 : Compléter l'ensemble Ω ci-dessous :

Ω = { (pierre , pierre) ; (pierre , feuille) ; (pierre , ciseau) ; (feuille , pierre) ; (feuille , feuille) , ……

Question 2 : Combien d'issues contient cet ensemble Ω ? : ……

Question 3 : Pour chacune des issues de Ω complété ci-dessus, colorier **ou** souligner en vert la figure qui gagne et en rouge celle qui perd.

Calcul des probabilités de gagner :

Si on suppose que le choix de la figure est réalisé d'une manière aléatoire, et que 2 joueurs A et B s'affrontent, on se propose de calculer les probabilités des évènements suivants :

- Evènement A : « le joueur A gagne »
- Evènement B : « le joueur B gagne »
- Evènement Nul : « match nul, A et B ont la même figure »

L'évènement A est ainsi un ensemble qui contient les issues suivantes :

A = {(pierre , ciseau) ; (feuille , pierre) ; (ciseau , feuille))

$$p(A) = \frac{nombre\ d'issues\ de\ A}{nombre\ d'issues\ de\ \Omega} = \frac{3}{9} = \frac{1}{3} \approx 0{,}33$$

Question 4 : Compléter

Compléter de même le calcul de probabilité pour l'évènement B :

B = {(ciseau , pierre) ; ……

$$p(B) = \frac{nombre\ d'issues\ de\ B}{nombre\ d'issues\ de\ \Omega} =$$

Compléter de même le calcul de probabilité pour l'évènement B :

Nul = {(pierre , pierre) ; ……

$$p(Nul) = \frac{nombre\ d'issues\ de\ Nul}{nombre\ d'issues\ de\ \Omega} =$$

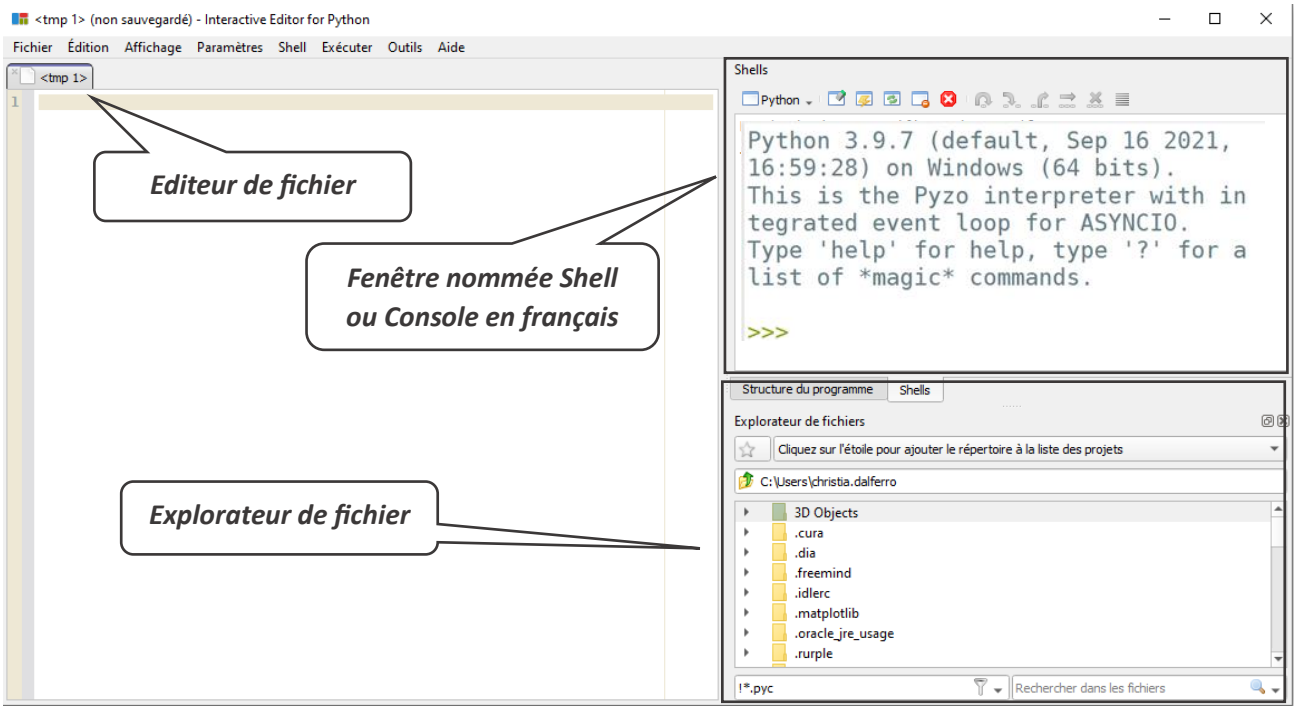
c) Simulation sur ordinateur : On se propose de simuler ici un jeu de Chifoumi en créant un code python.

⇨ Lancer le logiciel ***Pyzo*** en ouvrant le lien nommé « *Pyzo général* » qui se trouve dans le dossier « *Autres raccourcis* » se trouvant sur le bureau.

Dossier « *Autres raccourcis* »

Ce logiciel nommé ***Pyzo*** est ce qu'on appelle un **environnement de développement intégré** (I.D.E). Il vous permet de développer plus facilement des codes informatiques en langage Python. *Pyzo* propose à l'ouverture 3 fenêtres :

- **La console** ou shells sur Pyzo, permet :
  - d'exécuter des instructions Python individuellement,
  - de connaître l'état des variables,
  - d'afficher des messages pour l'utilisateur.
- **L'explorateur de fichiers** permet de visualiser le contenu du répertoire de travail
- **L'éditeur** permet d'écrire des lignes de code python dans un fichier qui pourront ensuite être toutes exécutées directement les unes après les autres.



⇨ Ecrire la ligne suivante **dans l'éditeur de fichier** :

```
from random import randint
```

Cette ligne permet d'importer la fonction randint().

⇨ Enregistrer ce fichier sous le nom chifoumi dans votre répertoire de travail sur U:/.

⇨ Appuyer sur la touche **F5** du clavier pour demander à Pyzo de lire les instructions écrites dans l'éditeur de fichier. Pour l'instant seule 1 ligne y a été écrite.

⇨ Pour se familiariser avec la fonction randint() qui a été importée, exécuter cette fois-ci dans la console, plusieurs fois `>>> randint(1,3)` . A chaque exécution, un nombre entier compris entre 1 et 3 est renvoyé.

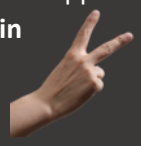
⇨ Exécuter à présent, **toujours dans la console** `>>> n = randint(1,3)`
Exécuter ensuite `>>> n` . Avec ces 2 instructions, le nombre aléatoire compris entre 1 et 3 est à présent stocké dans une variable nommée ici n. En exécutant `>>> n` dans la console, la valeur de cette variable est affichée.

⇨ Ecrire à présent dans l'éditeur de fichier, le script de la fonction main() définie ci-dessous :

def est un mot clé qui défini une fonction python

Ne pas oublier le double-point : après les parenthèses

main() est ici le nom donné à cette fonction. Comme on travaille sur une fonction qui simule un chifoumi, on l'appelle **main**



```
1  from random import randint
2
3  def main() :
4      n = randint(1,3)
5      if n == 1 :
6          return 'Pierre'
7      if n == 2 :
8          return 'Ciseau'
9      if n == 3 :
10         return 'Feuille'
```

Ne pas oublier le double-point :

4 espaces ou 1 tabulation de décalage

4 espaces ou 1 tabulation de décalage

⇨ Appuyer sur la touche F5 du clavier pour lire le fichier.

⇨ Exécuter à présent **dans la console**, cette fonction main() plusieurs fois.

A chaque exécution, cette fonction renvoie **aléatoirement** soit 'Ciseau', soit 'Feuille', soit 'Pierre'.

```
>>> main()
'Ciseau'

>>> main()
'Pierre'
```

```
1  from random import randint
2
3  def main() :
4      n = randint(1,3)
5      if n == 1 :
6          return 'Pierre'
7      if n == 2 :
8          return 'Ciseau'
9      if n == 3 :
10         return 'Feuille'
11
12 for n in range(1,10) :
13     A = main()
14     B = main()
15     print(A,B)
```

⇨ Compléter le code écrit dans l'éditeur en réalisant une **boucle** de 10 exécutions (instruction for n in range()). A chaque boucle, la fonction main() est exécutée deux fois. La première fois pour simuler la figure du joueur A, une seconde fois pour celle du joueur B. L'instruction print() permet d'afficher le résultat des figures de A et B, dans la console.

⇨ Appuyer sur la touche **F5** du clavier pour lire le fichier et exécuter le bouclage :

```
>>> (executing file "chifoumi.py")
Ciseau Feuille
Feuille Pierre
Pierre Ciseau
Pierre Feuille
Pierre Pierre
Pierre Feuille
Feuille Pierre
Pierre Ciseau
Pierre Pierre
```

Sur la figure ci-contre, on a rajouté des tests sur la valeur des variables A et B. Si le joueur A gagne le chifoumi, la variable nbA augmente sa valeur de 1. Même choses pour la variable nbB si c'est le joueur B qui gagne.

⇨ Le code qui s'exécute est incomplet. Il manque une série de test lorsque la valeur de A est égale à 'Ciseau'. Ecrire le code complet dans l'éditeur de fichier.

```
nbA = 0
nbB = 0

for n in range(1,10) :
    A = main()
    B = main()
    print(A,B)
    if A == 'Feuille' :
        if B == 'Pierre' :
            nbA = nbA + 1
        if B == 'Ciseau' :
            nbB = nbB + 1
    if A == 'Pierre' :
        if B == 'Feuille' :
            nbB = nbB + 1
        if B == 'Ciseau' :
            nbA = nbA + 1
    if A ==  ..............................
        if B == ..............................
            nbB = nbB + 1
        if B ==  ..............................
            nbA = nbA + 1
    print('A ',nbA/n,' B ',nbB/n)
```

Initialisation des variables nbA et nbB

Pour tester une égalité, on utilise le symbole ==

En appuyant sur la touche F5 du clavier, on obtient dans la console, quelque chose du type :

```
>>> (executing file "chifoumi.py")
Ciseau Ciseau
joueur A : 0.0  joueur B : 0.0
Feuille Ciseau
joueur A : 0.0  joueur B : 0.5
Ciseau Ciseau
joueur A : 0.0  joueur B : 0.3333333333333333
Ciseau Feuille
joueur A : 0.25  joueur B : 0.25
Pierre Pierre
joueur A : 0.2  joueur B : 0.2
Pierre Pierre
joueur A : 0.16666666666666666  joueur B : 0.16666666666666666
Feuille Pierre
joueur A : 0.2857142857142857  joueur B : 0.14285714285714285
Pierre Ciseau
joueur A : 0.375  joueur B : 0.125
Ciseau Pierre
joueur A : 0.3333333333333333  joueur B : 0.2222222222222222
```

les 2 figures jouées par les joueurs A et B

La proportion du nombre de parties gagnées par les joueurs A et B

Première boucle qui simule un premier chifoumi qui se termine par un match nul

⇨ Lorsque votre code fonctionne, le modifier pour réaliser un bouclage sur 1000 chifoumis simulés (toujours appuyer sur F5 pour lire le code de l'éditeur).

Question 5 : On a calculé dans la question 4, les probabilités des évènements A et B et on a trouvé que $p(A) \approx 0{,}33$ et $p(B) \approx 0{,}33$. Avec cette simulation de 100 chifoumis, retrouve-t-on ces proportions ?

⇨ Pour réaliser une simulation sur 1 000 000 de chifoumis, il est nécessaire de supprimer l'affichage print(A,B) dans la boucle et d'afficher les proportions des parties gagnées par les joueurs A et B uniquement lorsque le bouclage est terminé. Cela donne :

```
nbA = 0
nbB = 0

for n in range(1,1000000) :
    A = main()
    B = main()

    if A == 'Feuille' :
        if B == 'Pierre' :
            nbA = nbA + 1
        if B == 'Ciseau' :
            nbB = nbB + 1
    if A == 'Pierre' :
        if B == 'Feuille' :
            nbB = nbB + 1
        if B == 'Ciseau' :
            nbA = nbA + 1
    if A == 'Ciseau' :
        if B == 'Pierre' :
            nbB = nbB + 1
        if B == 'Feuille' :
            nbA = nbA + 1
print('joueur A :',nbA/n,' joueur B :',nbB/n)
```

print(A,B) supprimé

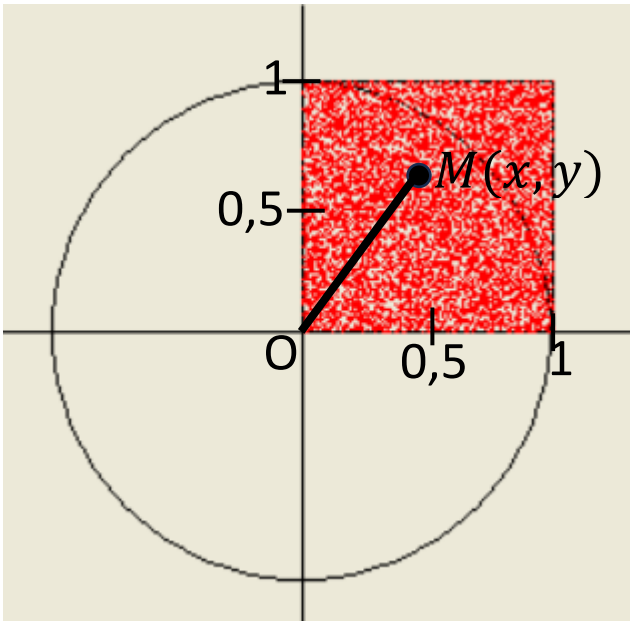
print( ) retiré de la boucle en décalant cette exécution sur le bord gauche de l'éditeur

Question 6 : On a calculé dans la question 4, les probabilités des évènements A et B et on a trouvé que $p(A) \approx 0{,}33$ et $p(B) \approx 0{,}33$. Avec cette simulation de 1 000 000 chifoumis, retrouve-t-on ces proportions ?

Question 7 : Cette simulation suppose que le choix de la figure est complètement aléatoire, un peu comme un lancer de dés. Or en réalité, le choix se fait plus en analysant les réactions de son adversaire. Lire l'article de la page https://fr.wikipedia.org/wiki/Pierre-feuille-ciseaux et écrire au verso de cette feuille, un texte de 10 lignes qui développe cette idée en argumentant.

EXERCICE 2. : PROBABILITES POUR TROUVER LA VALEUR DE π

Sur la figure ci-contre, on donne un cercle de centre $O(0,0)$ et de rayon 1. Plusieurs milliers de points M de coordonnées $M(x,y)$ ont été tracés en rouge. Les coordonnées x et y sont des nombres **déterminés aléatoirement entre 0 et 1**. Ces points se retrouvent donc tous dans un carré de coté 1, dont le coin bas, gauche est sur le point O.



Lorsque que les valeurs de x et y sont déterminées aléatoirement, le point M se retrouve à l'intérieur du cercle si $OM \leq 1$. Il se retrouve à l'extérieur du cercle si $OM > 1$. Cette distance est égale à $\sqrt{(x-0)^2+(y-0)^2}$, soit $OM = \sqrt{x^2+y^2}$.

Comme les points M se répartissent uniformément dans le carré, pour un nombre n de points tracés, la probabilité p que M se retrouve dans le cercle est égale à :

L'aire d'un disque est $\pi \times rayon^2$

$$p = \frac{aire\ du\ quart\ de\ cercle}{aire\ du\ carré} = \frac{\frac{\pi \times 1^2}{4}}{1 \times 1} = \frac{\pi}{4}$$

⇨ Ouvrir sur pyzo un nouveau fichier, à sauvegarder sous le nom calculPi.py .

⇨ Recopier le code donné ci-dessous. Le faire fonctionner avec la touche F5 du clavier. Le modifier pour qu'il puisse afficher en fin de boucle la valeur de $\pi \approx 3{,}14$. Laisser une trace de cette modification sur ce document.

```
1  from random import random
2  from math import sqrt
3
4  nb = 0
5  N = 100
6  for i in range(N) :
7      x = random()
8      y = random()
9      r = sqrt(x**2+y**2)
10     print(x,y,r)
11     if r < 1 :
12         nb = nb + 1
13
14 print(nb/N)
```

Importation de la fonction random() qui permet de définir un nombre aléatoire compris entre 0 et 1

(vous pouvez la tester en exécutant dans la console random())

Importation de la fonction racine carrée

Bouclage sur N valeurs