Feuille de TP 3: Simulations de lois discrètes, fonctions

L'objectif de ce TP est d'illustrer la notion de simulation aléatoire en `Python`. Dans tout le TP, on pensera à regarder dans l'aide pour comprendre ce que font les fonctions introduites.
Dans ce TP, on utilisera les bibliothèques suivantes :

```
import numpy as np
import matplotlib.pyplot as plt
```

```
import pandas as pan
import scipy.stats as st
```

1 Lois uniformes discrètes et conditionnement

Exercice TP3.1 On peut simuler des expériences (pseudo-aléatoires) sur un ordinateur, c'est-à-dire qu'un programme renvoie un résultat, qui, si répété, se comporte comme des tirages aléatoires indépendants. Par exemple, le programme suivant retourne une suite `D` de 1000 tirages de nombres entre 1 et 6, comme on aurait obtenu en lançant 1000 fois un dé.

```
N=1000
U=st.uniform.rvs(1,7,size=N)
D=np.floor(X)
```

On obtient donc une variable statistique comme si on avait fait une expérience aléatoire, et on peut donc la représenter comme une variable statistique discrète.

1. Calculer la table de contingence (des fréquences empiriques) de `D`.
2. Représenter `D` par un diagramme en bâton.
3. Que se passe-t-il si on augmente la taille N de l'échantillon ?
4. Créez une fonction `Dice(f,N)` qui prend en argument, un nombre de faces f et une taille d'échantillon, et qui renvoie la simulation d'un vecteur de N tirages d'un dé (équilibré) à f faces.
   Tester votre fonction pour un dé à 8 faces. (Vous vérifierez que sa table de fréquence empirique est bien approximativement uniforme comme souhaitée).

Exercice TP3.2

1. Créer un fonction `Restriction(x,k)` qui prend en argument un tirage x de nombres entiers et renvoie tous la suite des nombres de x inférieurs où égaux à k.
2. Soit `D= Restriction(Dice(8,N),6)`. Vérifier que c'est une simulation de loi uniforme.
3. Créer une fonction `Prime(x)` qui prend en argument un tirage x de nombres entiers et renvoie tous la suite des nombres premiers de x. On pourra utiliser `from sympy import isprime`. Soit `P= Prime(Dice(12,N))`. De quelle loi uniforme est-ce une simulation ?

2 Lois usuelles discrètes

Les lois usuelles sont dans la bibliothèque SciPy et plus précisément le paquetage stats. On les importe :

```
from scipy.stats import bernoulli, binom, geom, poisson
import scipy.stats as st
```

2.1 Diagrammes en bâtons

Exercice TP3.3

Pour tracer le diagramme en bâtons de la loi binomiale théorique $\mathcal{B}(10, 0.25)$, on peut utiliser la fonction `pmf` (*probability mass function*) ainsi :

```
n, p = 10, 0.25
ProbasB=binom.pmf(range(n+1),n,p)# probabilites de la distribution binomiale
plt.bar(range(n+1),ProbasB,width=0.05)
plt.show()
```

1. Tracer de même un diagramme en bâtons des lois $\mathcal{B}(10, 0.5)$, $\mathcal{B}(100, 0.25)$, $\mathcal{P}(2)$, $\mathcal{P}(10)$, $\mathcal{G}(0.75)$, $\mathcal{G}(0.25)$. On utilisera avec profit la page d'aide de SciPy.
2. Comparer à des diagrammes en bâtons de simulations de $\mathcal{B}(10, 0.5)$ et $\mathcal{P}(10)$, obtenues par `binom.rvs` ou `poisson.rvs`.
3. Nous voulons illustrer un théorème de convergence de la loi binomiale vers la loi de PoissonPour cela, représenter sur un même graphique les diagrammes en bâton des lois $\mathcal{B}(50, 0.2)$, $\mathcal{B}(250, 0.04)$, $\mathcal{B}(500, 0.02)$ et $\mathcal{P}(10)$. Quelle est la moyenne empirique de ses lois ? Que constatez-vous sur les diagrammes en bâton ?

2.2 Fonctions de répartition

Exercice TP3.4

1. Tracer la fonction de répartition [1] de la loi $\mathcal{B}(10, 0.25)$ (on pourra remplacer la fonction `pmf` par `cdf` (*cumulative distribution function*) et utiliser `step` au lieu de `bar`.)
2. Tracer de même la fonction de répartition de la loi binomiale $\mathcal{B}(100, 0.5)$ et de la loi de Poisson $\mathcal{P}(3)$. Que valent $\mathbb{P}(X \leq 3)$ et $\mathbb{P}(X > 30)$ si X suit la loi $\mathcal{B}(50, 0.2)$?

3 Supplément : Fonctions en Python et temps de calcul

Exercice TP3.5

1. Executer les commandes suivantes, et expliquer ce qu'elles font.

```
def r(x,y):
  z = np.sqrt(x**2+y**2)
  return(z)
r(3,4)
```

2. Écrire une fonction g qui à $N \in \mathbf{N}$ $p \in [0, 1]$ et $k \in [0, N]$ associe la probabilité qu'une variable aléatoire de loi Binomiale $\mathcal{B}(N, p)$ de paramètres N et p soit égale à k.

Exercice TP3.6

1. Exécuter les commandes suivantes, et expliquer ce qu'elles font.

```
from datetime import datetime
N = 10**4
t0 = datetime.now()
s = 0
for i in range(1,N):
  if (i%2==0):
    s = s+i**2
t1 = datetime.now()
t = np.arange(1,N)
s2 = np.sum(t[t%2==0]**2)
t2 = datetime.now()
print("Valeurs obtenues :\n s = ",s,"
    temps : ",t1-t0,"\n s2 = ",s2,"
   temps : ",t2-t1 ,"\n" )
```

2. Calculer la somme des inverses des carrés des 100000 premiers entiers impairs de deux façons. Au vue des temps de calcul, quelle sera la façon à privilégier à votre avis ?

1. C'est le nom de la fréquence cumulée pour une variable théorique.