

**TD : Simulations aléatoires 1**

*On n'oubliera pas d'importer la bibliothèque :*

*import numpy.random as rd*

1. On lance deux dés équilibrés à 6 face. On note X la variable égale au résultat du premier lancer, Y celle égale au second, Z égale à la somme des lancers, U égale au maximum des deux lancers, V égale à la distance entre les 2 numéros, W égale à 1 si les deux numéros sont égaux et 0 sinon.
  - a) Écrire un script qui permet de simuler X et Y.
  - b) Compléter le script pour simuler Z, U, V et W.
2. On lance une pièce une infinité de fois, la probabilité d'avoir pile est 1/2. On note X la variable égale au rang d'obtention du premier pile. On note Z la variable égale au rang d'obtention du second face.
  - a) Simuler X en utilisant une boucle while.
  - b) Simuler Z avec une boucle while.
  - c) On suppose que la probabilité d'avoir pile est égale à p. Modifier le script pour simuler Z.
3. On dispose d'une urne contenant n boules numérotées. On pioche sans remise jusqu'à obtenir le n°1. On note X la variable égale au nombre de tirage nécessaires.

Compléter et expliquer le script suivant permettant de simuler X

```
n=int(input('rentrer la valeur de n'))
X=1
while rd.random()<(n-1)/n:
    n=...
    X=...
```

4. On pioche des boules dans une urne contenant r boules rouges et b boules blanches. On pioche les boules sans remise jusqu'à avoir retiré toutes les boules blanches. On note X la variable aléatoire égale au nombre de tirage effectués, et Y la variable au nombre de boules rouges restantes.

Simuler X et Y à l'aide d'une boucle while (on pourra prendre r=10 et b=20).

5. Un mobile se déplace sur un axe gradué selon la règle suivante : à l'instant n il va en position 0,1,2,...,n de manière aléatoire uniforme. Il commence en position 0. On note la variable égale Xn sa position à l'instant n. On note Y la variable égale au rang du premier retour à la position 0.
  - a) Compléter la fonction suivante pour simuler une réalisation de Xn.

```
def SimulX(n):
    X=...
    return X
```

- b) Compléter la fonction pour qu'elle renvoie une simulation de Y.

```
def SimulY():
    n=1
    Y=1
    while SimulX(n) ... :
        Y=...
        n=...
    return ...
```

6. On pioche dans une urne qui contient 10 boules vertes et 5 boules rouges. Si on pioche une rouge on remet la boule dans l'urne et on ajoute 2 vertes, si on pioche une verte on remet la verte et on ajoute une boule rouge. On réalise de cette manière 10 tirages.
- Écrire un programme qui renvoie une simulation du nombre de boules rouges et vertes qu'on a à la fin de l'expérience.
  - En incorporant le programme précédant dans une boucle for, afficher une estimation du nombre moyen de boules rouges à l'issue d'une expérience.
  - Simuler la variable X qui est égale à 1 si on a plus de rouges à la fin que de verte, et 0 sinon.

7. *Loi binomiale négative.* Soit k un entier naturel fixé. On lance une pièce qui renvoie Pile avec probabilité p, jusqu'à obtenir k fois Pile. On note alors X la variable aléatoire égale au nombre de lancers effectués.

- a) Compléter le programme suivant pour qu'il affiche une simulation de X.

```
k=int(input('rentrez la valeur de l'entier k'))
c=0    #c compte les piles
X=0    # X compte le nombre de lancers
while ...:
    X+=1
    if ...:
        c+=1
print(X)
```

- b) Dans cette question on fixe k=2. Calculer la probabilité que la variable X soit égale à n, pour tout entier naturel n>1.