TP : Probabilité conditionnelle.

Vous travaillerez par groupe de 2.

On reprend l'énoncé du cours :

Énoncé : On dispose de deux urnes :

- L'urne U, avec 2 boules rouges et une boule noire.
- L'urne V, avec 1 boule rouge et une boule noire.

On lance un dé à 6 faces. Si l'on obtient :

- 1 (événement que l'on notera A) on tire une boule dans l'urne U.
- Sinon (événement que l'on notera $\bar{A}$), on tire une boule dans l'urne V.

On note :

- R l'évènement : "on obtient une boule rouge lors du tirage".
- N l'évènement : "on obtient une boule noire lors du tirage".

Pour simuler le lancé de dé à 6 faces :

```
import random as rd

def de():
    a=rd.randint(1,6)  # appel la fonction randint de la bibliothèque random
    if a==1:
        return 1     # on choisira un boule dans l'urne U
    else:
        return 2     # on choisira un boule dans l'urne V
```

Si l'on veut déterminer la fréquence d'apparition de "1" obtenu pour n simulation du lancé de dé :

```
def simulation_de(n):  # paramètre n
    nb_1=0
    for i in range(n):   # répétition n fois de l'expérience.
        sim_de=de()
        if sim_de==1:
            nb_1=nb_1+1
    return nb_1/n
```

1. Tester la fonction *simulation_de* pour $n = 10$ puis 100 puis 10000, puis commenter les résultats.
2. Reprendre l'implémentation de la fonction *de()* pour créer deux fonctions *Urne_U* et *Urne_V* pour simuler les tirages respectivement dans l'urne U et l'urne V. (on choisira d'affecter 0 pour le tirage d'une boule rouge et 1 pour le tirage d'une boule N.)
3. Créer une fonction *simulation_Urne_U* qui détermine la fréquence d'apparition d'une boule noire lors de n simulations de tirages avec remise dans l'urne U.

4. Voici l'implémentation complète :

```
   import random as rd

def de():
    a=rd.randint(1,6)  # appel la fonction randint de la bibliothèque random
    if a==1:
        return 1     # on choisira un boule dans l'urne U
    else:
        return 2     # on choisira un boule dans l'urne V

def Urne_U():   # simualation d'un tirage dans l'urne U
    simul=rd.randint(1,3)
    if simul==3:
        return 1  # on obtient 1 si la boule est noire
    else:
        return 0   # on obtient 0 si elle est rouge

def Urne_V(): # simualation d'un tirage dans l'urne V
    simul=rd.randint(1,2)
    if simul==2:
        return 1   # on obtient 1 si la boule est noire
    else:
        return 0   # on obtient 0 si elle est rouge

def exp():  # Simulation d'une expérience complète : tirage et boule
    if de()==1:
        return Urne_U()  # On otient 1 si la boule est noire
    else:
        return Urne_V()  # On otient 0 si la boule est noire

def simulation_exp(n):  # simualtion de n expériences complétes
    nb_N=0
    for i in range(n):
        nb_N=nb_N+exp()
    return nb_N/n  # fréquence d'apparition de boules noires

```

(a) Tester la fonction *simulation_exp(n)* pour vérifier le résultat trouvé en exercice.

(b) Modifier cette fonction pour vérifier le résultat obtenu pour $P(A \cap N)$.