

## Variables aléatoires en Python (Part. 2)

Pour manipuler des instructions ayant trait à l'aléatoire, on commencera toujours par importer la bibliothèque `np.random`. L'importation privilégiée est la suivante :

### Bibliothèque `numpy.random`

```
import numpy.random as rd
```

On a vu dans le TP précédent qu'on pouvait simuler la loi binomiale avec `rd.random`. En fait, on dispose d'instructions toutes-faites pour générer directement des valeurs selon une loi uniforme ou une loi binomiale :

### Simulation des lois finies usuelles

- `rd.randint(a,b+1)` génère un entier aléatoire selon la loi  $\mathcal{U}([a, b])$ .

On peut rajouter un paramètre à cette instruction pour générer plusieurs valeurs :

- `rd.randint(a,b+1,m)` génère un vecteur ligne de taille `m` contenant des réalisations indépendantes.
- `rd.randint(a,b+1,(m1,m2))` génère un tableau à `m1` lignes et `m2` colonnes contenant des réalisations indépendantes.

- `rd.binomial(n,p)` génère un entier aléatoire selon la loi  $\mathcal{B}(n, p)$ .

On peut rajouter un paramètre à cette instruction pour générer plusieurs valeurs :

- `rd.binomial(n,p,m)` génère un vecteur ligne de taille `m` contenant des réalisations indépendantes.
- `rd.binomial(n,p,(m1,m2))` génère un tableau à `m1` lignes et `m2` colonnes contenant des réalisations indépendantes.

### Exercice 1

Tester les instructions suivantes (plusieurs fois) et noter les résultats obtenus :

```
rd.randint(1,11) : .... rd.randint(1,11) : .... rd.randint(1,11) : ....
rd.binomial(5,0.5) : .... rd.binomial(5,0.5) : .... rd.binomial(5,0.5) : ....
```

```
rd.randint(0,5,4) : .....
```

```
rd.randint(0,5,(2,2)) : .....
```

### Exercice 2

"On lance 5 dés à 6 faces équilibrés".

Quelle instruction permet de simuler cette expérience aléatoire en Python ?

```
import numpy.random as rd
```

### Exercice 3

Un étudiant se confronte à un examen qui consiste en un QCM de 20 questions consécutives. Pour chaque question, il y a 4 réponses possibles.

L'étudiant répond complètement au hasard.

1. Définir une fonction `note` qui renvoie le nombre (aléatoire) de bonnes réponses données par l'étudiant.

```
def note() :
```

2. Adapter cette fonction pour que chaque bonne réponse rapporte 1 point, et chaque mauvaise réponse en retire 1.

```
def note() :
```

### Exercice 4

1. Proposer une fonction qui prend en entrée un entier  $n$  et qui, à l'aide d'une boucle `for`, renvoie le résultat (aléatoire) de la somme de  $n$  dés à 6 faces équilibrés.

```
def somme_de(n) :
```

2. Proposer une autre version, sans boucle, en utilisant `np.sum`.

```
def somme_de(n) :
```



### Exercice 5

(FLATHEAD game) Soit  $n \in \mathbb{N}^*$ .

- On génère un premier entier  $X$  choisi uniformément dans l'ensemble  $\llbracket 1, 2n \rrbracket$ .
- On génère un second entier  $Y$  forcément différent du premier :  
 $Y$  est choisi uniformément dans l'ensemble  $\llbracket 1, 2n \rrbracket \setminus \{X\}$ .

On propose alors le jeu suivant :

On montre à un joueur la valeur du premier entier  $X$ , et celui-ci doit prévoir si la valeur du second entier  $Y$  sera plus petite ou plus grande que  $X$ .

#### A - Implémentation du jeu en Python.

1. Quelle instruction Python permettrait de générer la valeur aléatoire  $X$  ?

```
x = .....
```

2. On suppose qu'on dispose de  $L = [x_1, x_2, \dots, x_p]$ , une liste de  $p$  valeurs. Compléter les instructions suivantes pour que  $Y$  soit une valeur aléatoire choisie uniformément dans l'ensemble  $\{x_1, x_2, \dots, x_p\}$ .

```
i = .....
Y = L[i]
```

3. L'instruction `L.remove(x)` permet de retirer la valeur  $x$  d'une liste  $L$ . Compléter le programme suivant pour qu'il simule le jeu présenté dans l'énoncé.

```
import numpy.random as rd
def game(n) :
    # Generation de la valeur X

    X = .....

    # Generation de la valeur Y

    L = list(range(.....))
    L.remove( .... )
    i = .....
    Y = L[i]

    # Interaction avec le joueur

    print("X = ",X)
    text = input("Y sera-t-il plus petit ou plus grand ? ")

    if (text == "petit" and Y < X) or (text == "grand" and Y > X) :
        print("Bravo !")
    else :
        print("Perdu !")

    print("Valeur de Y : ", Y)
```

Jouer quelques parties avec  $n = 5$  (donc des nombres générés entre 1 et 10).  
 On tapera ainsi `game(5)` dans la console.

#### B - Analyse mathématique du problème (à rédiger sur une feuille à part)

1. Quelle loi usuelle suit la variable aléatoire  $X$  ?
2. Pour tous  $i, j \in \llbracket 1, 2n \rrbracket$ , donner la valeur de la proba conditionnelle  $P_{[X=i]}(Y = j)$ .  
 (On distinguera les cas  $j = i$  et  $j \neq i$ ).
3. En déduire, pour tout  $j \in \llbracket 1, 2n \rrbracket$ , la valeur de  $P(Y = j)$ .  
 Quelle loi usuelle reconnaît-on ainsi pour la variable aléatoire  $Y$  ?
4. Pour tout  $i \in \llbracket 1, 2n \rrbracket$ , donner les valeurs des probas conditionnelles :

$$P_{[X=i]}(Y < X) \quad \text{et} \quad P_{[X=i]}(Y > X).$$

Selon la valeur de  $i$ , laquelle de ces probabilités est la plus grande ?

Selon la valeur de  $X$ , quelle est donc la stratégie que doit adopter le joueur s'il veut maximiser sa probabilité de faire la bonne prédiction ?

5. On suppose que le joueur adopte la stratégie optimale décrite dans la question précédente. On note  $A$  l'évènement "Le joueur fait la bonne prédiction".

Montrer que la probabilité de gagner au jeu est alors :  $P(A) = \frac{3n-1}{4n-2}$ .

On pourra décomposer le calcul en :

$$P(A) = \sum_{i=1}^n P([X=i] \cap A) + \sum_{i=n+1}^{2n} P([X=i] \cap A)$$

6. Lorsque  $n$  est très grand, quelle est donc, environ, la probabilité de gagner au jeu en adoptant la stratégie optimale ?