# TP 6 : Programmer l'aléatoire

2 BCPST 2 - TP Informatique

CONSEIL : à chaque nouvelle question faites une nouvelle cellule en tapant "#%%" .
Vous pourrez ainsi traiter toutes les questions dans un même fichier.

## Le module random

Le module `random` permet de générer des nombres élatoires. Il dispose de plusieurs fonctions, mais nous n'en utiliseront qu'une seule, la fonction `random()`

```
import random as rd

x = rd.random()
print(x)
--------------------
->0.16022955651881965
```

La fonction retourne un nombre $x \in [0,1[$ aléatoire et uniformément distribué dans $[0,1[$. Par exemple, $x$ a une chance sur 2 d'être inférieur à 0.5 et une chance sur trois d'être inférieur à $1/3$ etc...

## I Programmation de variables aléatoires :

Astuce : Dans ces questions, on pourra vérifier que le code fonctionne en estimant l'espérance obtenue sur les différentes lois programmées. Pour approcher une espérance, il suffit de faire la moyenne d'un grand nombre d'expériences.

**Q.1.** Programmer une fonction `bernoulli(p)` qui prend en entrée un paramètre $p \in [0,1]$ et qui renvoie 1 avec probabilité $p$ et 0 sinon.

**Q.2.** Programmer une fonction `binomiale(n,p)` qui prend en entrée un entier $n$ et paramètre $p \in [0,1]$ et qui renvoie une nombre $x \in [\![0,n]\!]$ suivant la loi binomiale. *Remarque : On pourra penser à une fonction récursive.*

**Q.3.** Programmer une fonction `géométrique(p)` qui prend en entrée un entier $n$ et paramètre $p \in [0,1]$ et qui renvoie une nombre $x \in \mathbb{N}^*$ suivant la loi géométrique de paramètre $p$.
**Q.4.** Programmer une fonction `uniforme(n)` qui prend en entrée un entier $n$ et qui renvoie une nombre $x \in [\![1,n]\!]$ suivant la loi uniforme.

**Q.5.** Programmer une fonction `uniforme2(a,b)` qui prend en entrée deux entiers $a$ et $b$ et qui renvoie une nombre $x \in [\![a,b]\!]$ suivant la loi uniforme.

**Q.6.** Programmer une fonction `poisson(lambda)` qui prend en entrée un paramètre flottant `lambda` strictement positif et qui renvoie une nombre $x \in \mathbb{N}$ suivant la loi de poisson de paramètre $\lambda > 0$.

## II Espérance, Variance et Loi

Pour la suite, on peut considérer n'importe quel algorithme codé précédemment afin d'évaluer l'espérence, la variance. On pourra prendre dans cette exemple une variable $X$ suivant la loi de poisson de paramètre 1.

**Q.7.** Ecrire un code permettant d'approcher l'espérance de $X$ en effectuant l'expérience un grand nombre de fois et en faisant la moyenne des résultats.

**Q.8.** De la même manière, faire une estimation de la variance de $X$.

**Q.9.** Donner une représentation de la loi de $X$ à l'aide d'un diagramme en bâtons. *On pourra utiliser la fonction* `plt.hist`.

## III Exercice 1 : les stickers

Dans l'album Panini de la coupe du monde 2026 de football masculin, on comptait en tout 48 équipes, ce qui a fait un nombre record de vignettes autocollantes à réunir pour compléter l'album. En effet, il faut en tout 980 autocollants pour que l'album soit complet.
Les autocollants de vendent en pochette. Une pochette contient 7 autocollants aléatoires.

Un fan de foot qui a beaucoup d'argent cherche à remplir son album Panini. Il achète donc des pochettes jusqu'à avoir complété l'album
**Q.10.** Coder une fonction `simulation1()` qui simule une variable aléatoire représentant le nombre de pochette achetée par le fan de foot afin de compléter son album.

**Q.11.** Effectuer une estimation de l'espérance du nombre de pochettes que le fan doit acheter pour finir son album. Sachant qu'une pochette coûte 1.20€, combien d'argent peut-il espérer dépenser ?

On suppose maintenant que deux fans de foot coopèrent pour compléter leurs albums. Ils achètent donc en même temps les pochettes. Ils prennent dans leur pochette respectives les autocollants qui leur manque, et lorsqu'ils ont déjà l'autocollant, ils le donnent à l'autre fan. Les deux fans s'arrêtent lorsque les deux albums sont complets.

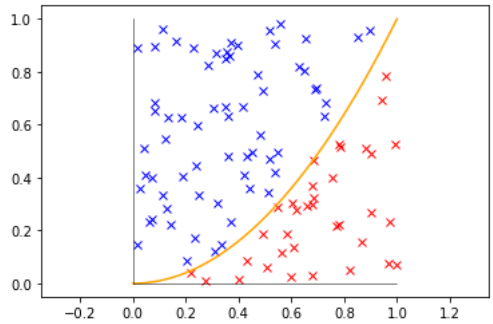
**Q.12.** Coder une fonction `simulation2()` qui simule une variable aléatoire représentant le nombre de pochettes achetée par les deux fans de foot afin de compléter leurs albums.

**Q.13.** Effectuer une estimation de l'espérance du nombre de pochettes que les fans doivent acheter pour finir les albums. Sachant qu'une pochette coûte 1.20€, combien d'argent vont-ils dépenser individuellement ?

# IV Exercice 2 : Méthode de Monte-Carlo

La méthode de Monte-Carlo est une manière propabiliste d'estimer l'aire sous la courbe d'une fonction. (donc d'estimer une intégrale)
Si l'on prend par exemple l'aire $\int_0^1 x^2 \, \mathrm{d}x = 1/3$, on peut "encadrer" la fonction dans le carré de [0,1] × [0,1] et effectuer des tirs uniformément aléatoire dans cette zone.
Ensuite c'est une loi de Bernoulli, si un point est sous la courbe, on le compte comme un succès, sinon c'est un échec. En moyenne, comme la surface sous la courbe représente un tiers de la surface du carré, on en déduit que l'aire sous la courbe de $f$ fait 1/3.

Illustration de la méthode de Monte-Carlo avec 100 points :




Sur ce graphique, tous les points ont été tirés aléatoirement et environs 1/3 des points sont sous la courbe, ce sont les points rouges.

**Q.14.** Ecrire un code python `Monte_Carlo(f,n)` qui prend en entrée une fonction `f` : $[0,1] \to [0,1]$ et un entier `n` qui retourne une approximation de l'intégrale $\int_0^1 f$ avec $n$ points aléatoires.
Tester cette fonction avec $f(x) = \sqrt{x}$ et $f(x) = \dfrac{1}{\sqrt{x}}$.

**Q.15.** Ecrire un code python `Graphique_Monte_Carlo(f,n)` qui prend en entrée une fonction `f` : $[0,1] \to [0,1]$ et qui permet de faire une illustration graphique de la méthode comme illustré par le graphe ci-dessus.

Maintenant, on va considérer la méthode pour calculer l'intégrale sur $[a,b]$. Pour le moment, on continue de considérer que la fonction est à valeur dans [0,1]. Sur $[a,b]$, la surface totale délimitée par l'expérience devient celle d'un rectangle.

**Q.16.** Ecrire un code python `Monte_Carlo(f,n,a,b)` qui prend en entrée une fonction `f` : $[a,b] \to [0,1]$ et qui retourne une approximation de l'intégrale $\int_a^b f$ avec `n` points aléatoires..
Tester cette fonction avec $\int_0^\pi \dfrac{\cos x + 1}{2} \, \mathrm{d}x$ et $\int_1^5 \dfrac{\mathrm{d}x}{x}$.

Enfin, pour avoir une étude complète, on va prendre une fonction $f$ continue sur $[a,b]$, qui n'est pas forcément à valeur dans $[0,1]$. On va considérer que $f$ est à valeur entre $[m,M]$, avec $m$ un minorant et $M$ un majorant de $f$ de sorte que $0 \in [m,M]$. Attention à bien penser à compter négativement les aires de la courbe lorsqu'elle est en dessous de l'axe $(Ox)$.

**Q.17.** Ecrire un code python `Monte_Carlo(f,n,a,b,m,M)` qui prend en entrée une fonction `f` : $[a,b] \to [m,M]$ et qui retourne une approximation de l'intégrale $\int_a^b f$ avec `n` points aléatoires..
Tester cette fonction avec $\int_0^2 x^2 \, \mathrm{d}x$, $\int_{-\pi}^{\pi} \sin x \, \mathrm{d}x$ et $\int_{-10}^{10} \dfrac{\sin x}{x} \, \mathrm{d}x$.

# V Covariance

On lance indépendamment deux dés équilibrés à six faces. On note $X$ la somme des deux dés et $Y$ le produit des deux dés.
**Q.18.** Coder une fonction `simul()` qui retourne `X,Y` les valeurs de `X` et de `Y`.

**Q.19.** Effectuer graphiquement un nuage de point représentant plusieurs simulations de l'expérience en plaçant $X$ en abscisse et $Y$ en ordonnée. Que remarque-t-on ? Que peut-on deviner concernant la covariance ?

**Q.20.** Ecrire un code permettant d'approcher la covariance entre $X$ et $Y$ en effectuant l'expérience un grand nombre de fois.

**Q.21.** Ecrire un code permettant d'approcher le coefficient de corrélation linéaire entre $X$ et $Y$ en effectuant l'expérience un grand nombre de fois.