La méthode de Monte-Carlo

C'est quoi ?

La méthode de Monte-Carlo consiste à approcher une solution d'une équation, ou toute valeur numérique, en utilisant des expériences aléatoires et donc les probabilités. Le nom de cette méthode fait référence aux jeux de hasard du quartier de Monte-Carlo (à Monaco). Cette méthode a été inventée en 1947 par Nicholas Metropolis, et publiée en 1949 dans un article scientifique avec Stanislaw Ulam lors du développement de l'arme nucléaire.

Nous allons nous en servir ici pour donner une estimation du nombre irrationnel π.

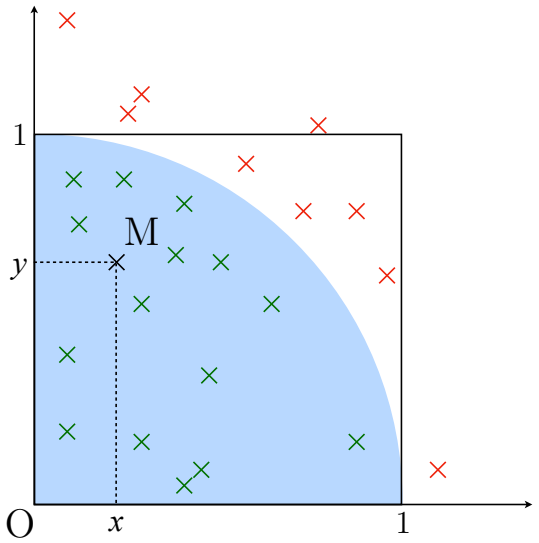
Méthode

Nous allons tracer un carré de côté 1. Dans ce carré, nous allons générer des points de façon uniforme mais aussi de façon aléatoire.
Nous allons également tracé le quart de disque de rayon 1 dont le centre de ce cercle correspond au sommet inférieur gauche du carré. On va alors calculer la proportion de points situés à l'intérieur de ce quart de disque afin d'avoir une estimation de son aire, qui est $\frac{\pi}{4}$.

$$\pi \approx \frac{\text{nombre de points à l'intérieur du quart de disque}}{\text{nombre de points à l'extérieur}}$$

Equation cartésienne d'un cercle de centre $(0\,;\,0)$ et de rayon 1 : $x^2 + y^2 = 1$

On a donc $OM = x^2 + y^2$
mais avec $x^2 + y^2 < 1$ pour être à l'intérieur du cercle.



Nous allons utiliser la fonction random de Python. Elle permet de générer un nombre aléatoire entre 0 et 1.

```
1  from random import random as rd
2  n = 1000
3  s = 0
4  for i in range(n):
5      x = rd()
6      y = rd()
7      if ... :
8          s=s+1
9  estimation = ...
10 print(estimation)
```

1. Que représente la variable s ?

   .....

   .....

   .....

2. Compléter les lignes 7 et 9 du programme.

3. Tester le programme en générant 1 000 points.

   .....

   On va maintenant stocker les différentes estimations obtenues dans une liste tout en faisant varier le nombre de points générés.

```
1  matplotlib inline
2  import matplotlib.pyplot as plt
3  N=500
4  estimation = []
5  somme = 0
6  for i in range(N):
7      (x,y) = (rd(),rd())
8      if x**2+y**2<1:
9          somme = somme + 1
10     estimation.append(...)
11
12 plt.plot([0,N],[3.14,3.14])
13 plt.text(N+2,3.13,"$\pi$")
14 plt.title("Estimation de $\pi$")
15 plt.plot(estimation)
16 plt.show()
```

4. Que contient la liste estimation ?

   .....

   .....

   .....

5. Compléter la ligne 10 du programme.