1. Méthode de Monte-Carlo

On cherche une représentation de l'inéquation [1] :

$$(x^2+y^2-1)^3 < 4x^2y^3$$

```
# -*- coding:utf8 -*-
# python 3
#
# on cherche une représentation de
# (x^2 + y^2-1)^3 < 4 x^2 y^3
# utilisation de la méthode de Monte-Carlo

from random import uniform
from matplotlib import pyplot as plt

for i in range(5000):
    x, y = uniform(-2,2), uniform(-1.5,2)
    if (x**2 + y**2-1)**3 < 4*(x**2)*(y**3):
        plt.plot(x,y,'ro') # point rouge si OK
    else:
        plt.plot(x,y,'b.') # pixel bleu sinon

plt.title("une belle inégalité")
plt.axis('equal')
plt.axis('on')
plt.show()
```

1 – et on n'a pas pensé à utiliser GeoGebra ?!?

2. Spaghetto

On coupe un spaghetto en 3 morceaux : quelle est la probabilité que ces trois morceaux soient les côtés d'un triangle ?

- Première modélisation : on coupe deux fois simultanément au hasard le spaghetto.
- Seconde modélisation : on coupe une première fois au hasard, on obtient donc deux morceaux. Le plus grand est coupé une seconde fois au hasard.

Pour chacune des modélisations on veut simuler quelques milliers de coupes, puis obtenir un visuel de la probabilité en utilisant la méthode de Monte-Carlo : x et y sont des réels de $[0\,;1]$ ils représentent l'abscisse de chacune des coupes (le spaghetto a une longueur de 1).

- module matplotlib
- module random fonction uniform
- test - max

accélération de l'affichage en utilisant des listes