

Sciences Numériques et Technologie

Ressources dans <https://github.com/PhilippeRenavierGonin/snt>

Philippe.Renavier@ac-grenoble.fr

Introduction à Python...

Bilan de la 1^{re} activité

Ouvrez Thonny, pour essayer certaines choses en parallèle

Comparatif Scratch / Python

- (il existe des outils pour traduire des programmes exprimés en Scratch en Script Python)
- Implémentation de la même chose :
 - <https://scratch.mit.edu/projects/1049194930/>
 - Et le fichier PI_MonteCarlo.py
- Note : on parle de « script python » car c'est interprété (c'est le logiciel python qui exécute le code, ce n'est pas compilé)

Où commence le programme

```
import math
import random
```

Point d'entrée dans le script,
parfois masqué par les outils

```
if __name__ == '__main__':
```

```
    nbTentatives = 10100
```

```
    nbReussites = 0
```

```
    for i in range(1, nbTentatives+1):
```

```
        x = random.uniform(0,2)
```

```
        y = random.uniform(0,2)
```

```
        distance = math.sqrt((x-1)*(x-1)+(y-1)*(y-1))
```

```
        if distance <= 1:
```

```
            nbReussites = nbReussites + 1
```

```
    pi = 4*nbReussites/nbTentatives
```

```
    print(f"approximation de pi = {pi}")
```

quand ce sprite est cliqué

*Il faut exécuter dans le logiciel, en ligne
sur un site web ou en ligne de commande
« python nom_du_fichier.py »*

Variable

```
import math
import random
```

```
if __name__ == '__main__':
```

```
    nbTentatives = 10100
```

```
    nbReussites = 0
```

```
    for i in range(1, nbTentatives+1):
```

```
        x = random.uniform(0,2)
```

```
        y = random.uniform(0,2)
```

```
        distance = math.sqrt((x-1)*(x-1)+(y-1)*(y-1))
```

```
        if distance <= 1:
```

```
            nbReussites = nbReussites + 1
```

```
    pi = 4*nbReussites/nbTentatives
```

```
    print(f"approximation de pi = {pi}")
```

Définir et affecter une valeur
à des variables



Ici, ces variables sont des nombres entiers,
appelés « int », abréviation de « integer »

Le type des variables dépend des affectations

- Affecter : donner une valeur
- Variable : « symbole » qui représente une valeur changeante
 - à l'initialisation
 - en cours d'exécution
- Nom de Variable : explicite, pour savoir ce que c'est
 - il y a des mots clefs utilisés par python : on ne peut pas s'en servir autrement
 - Par exemple, on ne peut pas appeler une variable « if ».
- Type « courant » des variables :
 - Nombre entier (int)
 - Nombre « réel » (float)
 - Expression booléenne (boolean, True / False)
 - Chaine de caractères (string)

Essayons avec :

```
a = 1
print(a)
print(type(a))
# et refaisons ceci avec :
# a = 1.0
# a = True
# a = "un message"
```

Boucle « for » : pour différentes valeurs, refaire les mêmes actions

```
import math
import random
```

```
if __name__ == '__main__':
    nbTentatives = 10100
    nbReussites = 0
    for i in range(1, nbTentatives+1):
        x = random.uniform(0,2)
        y = random.uniform(0,2)
        distance = math.sqrt((x-1)*(x-1)+(y-1)*(y-1))
        if distance <= 1:
            nbReussites = nbReussites + 1
    pi = 4*nbReussites/nbTentatives
    print(f"approximation de pi = {pi}")
```

Une boucle for pour répéter des actions un nombre de fois déterminé



Même si « i » (pour indice) n'est pas utilisé ici, i prend toutes les valeurs entre 1 (inclus) et « nbTentatives+1 » (exclus) et pour chaque valeur possible de « i », ce qui est dans la boucle est exécuté

Délimitation par l'indentation

- Indentation : espace entre le côté gauche et le début des caractères
- Ici, on a 3 niveaux :

Niveau 1 :
dans le point d'entrée

```
nbTentatives = 10100
nbReussites = 0
for i in range(1,nbTentatives+1):
    # [...]
pi = 4*nbReussites/nbTentatives
print(f"approximation de pi = {pi}")
```

Niveau 2 :
dans la boucle for

```
x = random.uniform(0,2)
y = random.uniform(0,2)
distance = math.sqrt((x-1)*(x-1)+(y-1)*(y-1))
if distance <= 1:
    # [...]
```

Niveau 3 :
dans le « if »

```
nbReussites = nbReussites + 1
```


L'indentation

Niveau 1 : dans le point d'entrée

Niveau 2 : dans la boucle for

Niveau 3 : dans le « if »

```
import math
import random

if __name__ == '__main__':
    nbTentatives = 10100
    nbReussites = 0
    for i in range(1, nbTentatives+1):
        x = random.uniform(0,2)
        y = random.uniform(0,2)
        distance = math.sqrt((x-1)*(x-1)+(y-1)*(y-1))
        if distance <= 1:
            nbReussites = nbReussites + 1
    pi = 4*nbReussites/nbTentatives
    print(f"approximation de pi = {pi}")
```

Le retour à une indentation précédente mais fin au « if »

Variable, de type « réel » dit « float »

```
import math
import random
```

```
if __name__ == '__main__':
    nbTentatives = 10100
    nbReussites = 0
    for i in range(1, nbTentatives+1):
```

```
        x = random.uniform(0,2)
        y = random.uniform(0,2)
        distance = math.sqrt((x-1)*(x-1)+(y-1)*(y-1))
```

```
        if distance <= 1:
```

```
            nbReussites = nbReussites + 1
```

```
pi = 4*nbReussites/nbTentatives
```

```
print(f"approximation de pi = {pi}")
```

Définir et affecter
une valeur à des
variables



Ici, ces variables sont des nombres « réels », appelés « float », pour « virgule flottante »
random est un module qui permet de manipuler des nombres « aléatoires » (ou pseudo aléatoire)
math est un module qui offre des fonctions mathématiques, ici racine carré (square root, sqrt)

Opérations mathématiques

- Addition : $a + b$
- Soustraction : $a - b$
- Multiplication : $a * b$
- Division (le résultat est un «réel») : a / b
 - $10 / 4$ vaut 2.5
- Division entière : $a // b$
 - $10 // 4$ vaut 2
- Reste de la division entière (modulo) : $c \% b$
 - $11 \% 4$ vaut 3
- Puissance : $a ** b$
 - $2 ** 10$ vaut 1024

Essayons avec :

```
a = 10  
b = 4  
c = 11  
# exemple  
print(a+b)
```

Documentation en ligne

- Par exemple, pour random :
<https://docs.python.org/3.12/library/random.html#random.random>
- Il y a la documentation officielle de python
- Il y a la documentation des modules
 - Certains modules sont officiels, d'autres non
 - Module : élément qui fournit des « fonctions » non nécessaires à chaque exécution
 - À utiliser sur demande, avec « import » en début de script

Instruction conditionnelle « if » : si (condition) alors (actions) sinon (actions)

```
import math
import random

if __name__ == '__main__':
    nbTentatives = 10100
    nbReussites = 0
    for i in range(1, nbTentatives+1):
        x = random.uniform(0,2)
        y = random.uniform(0,2)
        distance = math.sqrt((x-1)*(x-1)+(y-1)*(y-1))
        if distance <= 1:
            nbReussites = nbReussites + 1
    pi = 4*nbReussites/nbTentatives
    print(f"approximation de pi = {pi}")
```

Forme avec si (condition) alors et l'alternative sinon (si la condition est fausse)
if (expression booléenne):
 # code pour le cas où l'expression est vraie
else:
 #code pour le cas où l'expression est fausse (partie non obligatoire)

Définir et affecter
une valeur à des
variables



Les « expressions booléennes » sont des conditions :
 mathématiques (==, !=, <, >, <=, >=) ou
 des booléens : True (vrai) et False (faux) et d'autres expressions entre booleén

Expressions entre booléen

- Booléen (boolean) : une valeur qui vaut soit vrai (True) soit faux (False)
- Soit a, b deux booléens
 - a and b : (and = et) expression qui est vrai uniquement si a vaut vrai et si b vaut vrai
 - Si a ou si b vaut faux ou les deux, alors a and b vaut faux
 - a or b : (or = ou) expression qui est vrai si a vaut vrai ou si b vaut vrai ou les deux valent vrais.
 - Si a vaut faux et si b vaut faux, alors a or b vaut faux
 - not a : (not = négation, le contraire) expression qui vaut vrai si a est faux. not a est le contraire de a

print : afficher du texte dans une console (textuelle)

```
import math
import random

if __name__ == '__main__':
    nbTentatives = 10100
    nbReussites = 0
    for i in range(1, nbTentatives+1):
        x = random.uniform(0,2)
        y = random.uniform(0,2)
        distance = math.sqrt((x-1)*(x-1)+(y-1)*(y-1))
        if distance <= 1:
            nbReussites = nbReussites + 1
    pi = 4*nbReussites/nbTentatives
    print(f"approximation de pi = {pi}")
```

Affichage de la valeur de la variable pi

montrer la variable pi ▼

La console : une zone de texte, généralement en bas des logiciels d'édition et d'exécution de python, ou à droite dans les pages web d'édition en ligne de python

Entrée/Sortie en Python

Input/Output (i/o ou io ou IO) en anglais

Ouvrez Thonny, pour essayer en parallèle

print avec un peu plus d'information

- Affiche dans la console un message et termine par un retour à la ligne
- Différentes formes
 - `print(variable1, variable2, variable3)` : affiche la valeur de chaque variable mise en paramètre avec un espace entre chaque valeur
 - `print("une chaine de caractères")` : pour afficher littéralement ce qui est entre les guillemets
 - `print(f"une chaine de caractères, variable1 = {variable1}")` : le f avant le 1^{er} guillemet permet de dire que des variables seront dans la chaine de caractère. Ces variables (ou expression) sont délimitées par des accolades { }
 - Essayez dans la continuité des précédents essais
- `\` permet d'insérer des caractères particuliers dans une chaine de caractères :
 - `\"` permet d'avoir des guillemets dans la chaine de caractères
 - `\n` permet d'avoir un retour à la ligne, `\t` une tabulation
 - Etc.

Saisir des valeurs en entrée

- `input("avec un message expliquant ce qui est attendu")` : permet de demander à l'utilisateur de taper une valeur
- Ce qui est obtenu, c'est une chaîne de caractères.

- Conversion de type :

- `int()`, `float()`, `str()`
- Essayez avec des valeurs fixes
- Puis essayez avec `input`

```
a = "10"                                # ou a = "10.5"
b = int(a)                              # ou b = float(a)
print(a, type(a), b, type(b))
c = 11
d = str(c)
print(c, type(c), d, type(d))
a = input("entrez un nombre : ")
print(a, type(a))
a = int(input("entrez un nombre : "))
print(a, type(a))
```

Lire depuis un fichier

Écrire dans un fichier

- Cela sera abordé au fur et à mesure
- open : permet d'ouvrir un fichier, en précisant le mode :
 - « lecture seule » : r (read)
 - « écriture (écrasement) » : w (write)
 - « écriture en fin de fichier » : a (append)
- On verra un exemple dans l'activité 2