

Expérimentation de la classe inversée pour l'enseignement de la programmation Python à l'université

Pierre Poulain

pierre.poulain@u-paris.fr
@pierrepo

CL!C
2020

1^{er} novembre 2020





Une classe inversée (sans vidéo) pour apprendre à programmer (*presque* sans ordinateur)

Bonjour !

Maître de conférences
à l'Université de Paris

Chercheur en bioinformatique



Au menu

Pourquoi ?

Comment ?

Et alors ?

Et après ?

Organisation du cours

Cours « Programmation Python 1 »

29 étudiants de master 1 (bio- & chimie-informatique)
débutants en programmation

Septembre à décembre 2019 (avant Covid-19) :

- **7 séances de cours magistraux (CM) de 2 h**
- 8 séances de travaux pratiques (TP) de 2 h, en salle machine

Accompagnement



Antoinette Bouziane



Marine Lanteri

Service d'Accompagnement aux Pédagogies Innovantes et à
l'Enseignement Numérique de Sorbonne Paris Cité (SAPIENS)

<https://sapiens-uspc.com/>



Pourquoi ?

Point de départ

+10 ans d'enseignement en Python

Formation à la pédagogie (SAPIENS)

Envie de recentrer les sessions de cours magistraux sur les étudiants.

Diaporama → *live coding* → inversion

Difficultés de cet enseignement

« Programmation Python » =

1. Concepts universels de programmation (variable, boucle, test...)
2. Le langage « Python »
3. Manipulation d'un ordinateur (*computer literacy*)

Matériel de cours

Classe inversée de type I
l'enseignant fournit le support de cours
(vidéo, poly...)

Matériel de cours mature

<https://python.sdv.univ-paris-diderot.fr/>

disponible sous licence CC BY-SA





Comment ?

Activités en amont

Lecture des chapitres de cours (poly + en ligne).

Rédaction d'une « fiche mémo » = un résumé par chapitre (A4, recto, manuscrit) à déposer sur la plateforme en ligne (Moodle) de l'Université.

Activités en cours magistraux

Principe général :
développer des compétences
en programmation Python
mais **sans ordinateur**

Fiches mémo (2)

Comparaison et correction par les pairs des fiches mémo (10')

- Erreurs
- Oublis
- Exemples

Fiches mémo (3)

Chapitre 7: les fichiers

Léo B

bruce.txt = *qualité*
ca. fait
jeu. court!

n: lecteur seule

ouvrir le fichier

fichier = objet type *fichier ouvert*

close = fermer le fichier

filin = open("bruce.txt", "r")
fichier.readline()

relut la, 'ca. fait' \n, 'jeu. court' \n → liste des lignes du fichier!

filin.close()

⚠ Impossible de lire un fichier fermé

READLINE()

WITH open("bruce.txt", "r") as filin:

lignes = filin.readlines()

for ligne in lignes:

print(ligne)

FORMATION AUTO

relut
ca. fait
jeu. court!

WITH

WITH open("bruce.txt", "r") as filin:

filin.read()

relut tout le fichier en une seule fois → chaîne de caractères!

READ()

WITH open("bruce.txt", "r") as filin:

lignes = filin.readlines()

while ligne != "":

print(ligne)

lignes = filin.readlines()

relut
ca. fait
jeu. court!

Suivre son ligne

READLINE()

OUVRIR DEUX FICHIERS

WITH open("jeu.txt", "r") as fichier1, open("qualite.txt", "r") as fichier2:

for ligne in fichier1:

fichier2.write("x " + ligne)

Donc jeu.txt =

* qualité

* court

CH15: Bonnes pratiques en Python

Léo B

① Indentation

→ 4 espaces

② Modules

→ plutôt importer module que from module import *

③ Nommage

* variables:

ma_variable

* constantes

MA_CONSTANTE

* classes

MaClasse

④ Espaces

* entiers (+, -, /, *, **, %, //, >, <, >=, <=, and, or, ...)

* 4 espaces de {}, {} et {}

* après :, mais pas avant

⑤ Commentaires

xx = xx # In english please

⑥ Docstrings

""" Docstring simple """

ou

""" Docstring long """

Se démarquer les détails

~~~~~

Se resumer!

"""

⑦ Qualité: contrôle qualité

\* pydocstyle: conforme avec la PEP 8?

↳ & pydocstyle --help --help --help

→ donne un rapport

\* pydocstyle: conforme avec la PEP 257 et docstrings

↳ donne un rapport

\* pylint: tente de compiler le code et propose des améliorations

⑧ Organiser son code

① Docstring décrivant global le script

② Métadonnées (préférences)

③ Modules (internes, puis externes)

④ Constantes

⑤ Classe

⑥ Docstring décrivant la classe

⑦ Signatures avant chaque méthode de classe

⑧ Fonctions classiques, 2 lignes vides avant

⑨ Programme principal

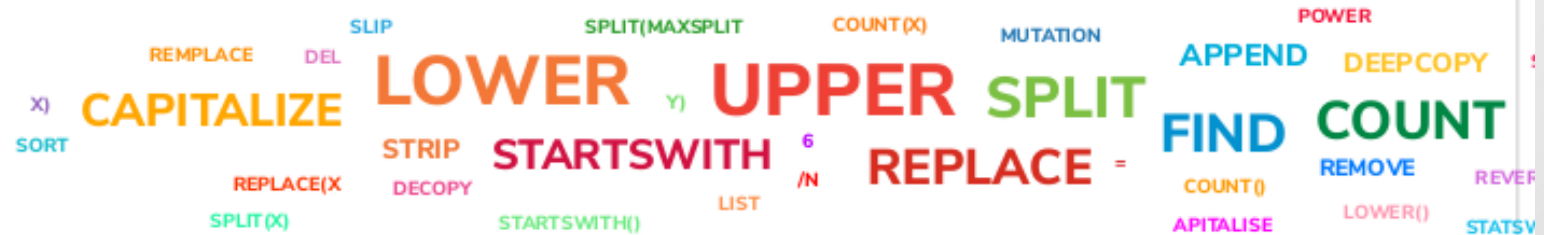
LISTES

# Quiz en ligne (WooClap)

## Question ouverte

1

Listez les méthodes disponibles pour les chaînes de caractères.  
Sans point ni parenthèse.



A word cloud of Python string methods. The words are in various colors and sizes, arranged in a horizontal, slightly overlapping manner. The most prominent words are 'LOWER' in large orange letters, 'UPPER' in large red letters, 'SPLIT' in large green letters, 'FIND' in large blue letters, and 'COUNT' in large green letters. Other visible words include 'CAPITALIZE' (orange), 'REPLACE' (red), 'STARTSWITH' (red), 'STRIP' (red), 'REVERSE' (red), 'APPEND' (blue), 'DEEPCOPY' (yellow), 'POWER' (red), 'MUTATION' (blue), 'SPLIT(MAXSPLIT)' (green), 'COUNT(X)' (orange), 'REPLACE(X)' (red), 'SPLIT(X)' (green), 'REVERSE' (red), 'REMOVE' (blue), 'APITALISE' (purple), 'LOWER()' (red), 'STATSV' (blue), 'LIST' (orange), 'STARTSWITH()' (green), 'DECOPY' (red), 'X)' (purple), 'SORT' (blue), 'DEL' (purple), 'Y)' (green), '6' (purple), and '/N' (orange).



# Quiz en ligne (WooClap)

## QCM

2

Que renvoie le script toto.py ?

|                                                      |                        |     |          |
|------------------------------------------------------|------------------------|-----|----------|
| <code>['toto.py', 1, 8, 'souris', 3.14]</code>       | <div><div></div></div> | 0%  | 0 votes  |
| ✓ <code>['1', '8', 'souris', '3.14']</code>          | <div><div></div></div> | 35% | 7 votes  |
| <code>['toto.py', '1', '8', 'souris', '3.14']</code> | <div><div></div></div> | 55% | 11 votes |
| <code>[1, 8, 'souris', 3.14]</code>                  | <div><div></div></div> | 10% | 2 votes  |

# Script à trous : principe

Mise en situation :

- Programme (script) fonctionnel
- Écrit par le prof
- Avec des trous

Les étudiants remplissent les trous :

- Compréhension et **adaptation** au contexte
- Pas de page blanche

# Script à trous : exemple

## La spirale : programmez-la !

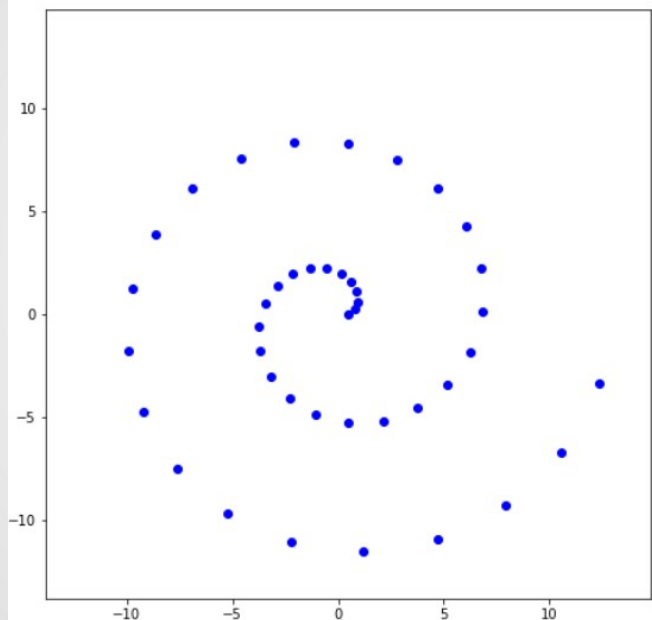
$\theta : 0 \rightarrow 4\pi$  par pas de 0.1  
 $r : 0.5 \rightarrow \dots$  par pas de 0.1

Utilisez le module `math`, ainsi que les fonctions `math.cos()` et `math.sin()` et la constante `math.pi`.

Enregistrez les coordonnées cartésiennes dans le fichiers `spirale.dat` en respectant le format suivant :

- un couple de coordonnées  $(x_A, y_A)$  par ligne ;
- un espace entre les deux coordonnées  $x_A$  et  $y_A$  ;
- les coordonnées affichées sur 10 caractères avec 5 chiffres après la virgule.

Affichez ensuite la spirale obtenue avec le module `matplotlib`.



# Script à trous : exemple

La photocopieuse a « mangé » certaines zones du script. Essayez de les compléter.

```
1 import 
2 with open(, ) as :
3     theta = 
4     rayon = 
5      = 0.1
6     while theta < :
7         x = (theta) * 
8         y = (theta) * 
9         .write("{} {}\n".format(, ))
10        theta += 
11        rayon += 
12
13 import matplotlib.pyplot as plt
14 # Pour afficher la spirale, il faut une liste qui contient
15 # les coordonnées x et une autre les coordonnées y
16  = []
17  = []
18 with open(, ) as f_in:
19     for line in f_in:
20         coords = line.split()
21         .append((coords[0]))
22         .append((coords[1]))
23 plt.figure(figsize=(8,8))
24 mini = min(x+y) * 1.2
25 maxi = max(x+y) * 1.2
26 plt.xlim(mini, maxi)
27 plt.ylim(mini, maxi)
28 plt.plot(x, y)
29 plt.savefig("spirale.png")
```

 Pierre Poulain  
Document sous licence Creative Commons Attribution 4.0 International (CC BY 4.0)

La photocopieuse a « mangé » certaines zones du script. Essayez de les compléter.

```
1 import 
2 with open(, ) as :
3     theta = 
4     rayon = 
5      = 0.1
6     while theta < :
7         x = (theta) * 
8         y = (theta) * 
9         .write("{} {}\n".format(, ))
10        theta += 
11        rayon += 
12
```

# Script tournant : principe

Inspiré des « tableaux tournants » de JC Cailliez  
(mais linéaire et individuel)

1. Mise en situation.
2. Un étudiant débute la rédaction du programme.
3. Un second poursuit.

Innovez dans votre classe... faites tourner les tableaux !

JC Cailliez, 09/02/2020

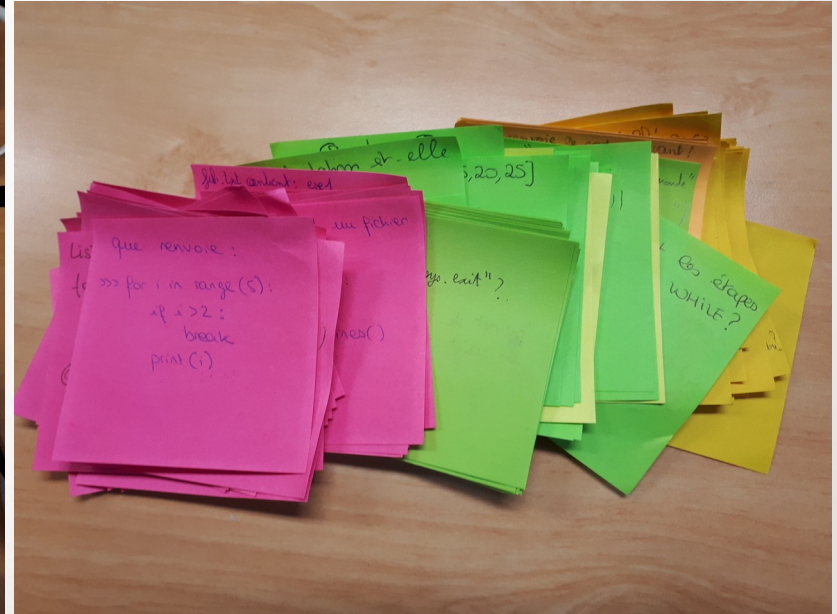
<http://blog.educpros.fr/jean-charles-cailliez/2020/02/09/innovez-dans-votre-classe-faites-tourner-les-tableaux/>

# Question suivante : principe

Préparation à un examen de QCM.

1. Une fiche : recto (question) / version (réponse).
2. Tri / affinage intra-groupe.
3. Évaluation inter-groupe.

# Question suivante : illustration





# Question suivante

## CONSEILS

- GESTION DU TEMPS
- MILIEU OU FIN DE SEMESTRE
- DST-IT





# Activités en travaux pratiques

Utilisation d'ordinateurs (enfin).

Mise en application des notions découvertes à la maison et développées en cours.

Travail en binôme « *pair programming* » : la programmation comme activité sociale.

# Évaluation sommative

QCM

Production de programmes pour répondre à des mises en situation authentiques :

- Projet fil rouge : Rosalind (<http://rosalind.info/>)
- Examen final








# Et alors ?

# Évaluation de l'enseignement (EEE)

Comment avez-vous apprécié les activités suivantes ?

1 : pas du tout, 5 : beaucoup

|                                                                                                |                                                                                       |     |
|------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------|-----|
| Les fiches mémo.                                                                               |    | 3.8 |
| Les associations d'idée ou de concept.                                                         |    | 3.6 |
| Le script tournant (vous commencez à écrire un script puis vous l'échangez avec votre voisin). |    | 4.3 |
| Le script à trous (l'imprimante a "mangé" certaines parties du script).                        |    | 4.4 |
| La question suivante (vous concevez des questions de QCM que vous posez ensuite aux autres).   |    | 4.2 |
| Les quiz et nuages de mots sur WooClap.                                                        |  | 4.4 |

# Évaluation de l'enseignement (EEE)

Je pense qu'il faudrait donner une "fiche squelette" pour faire les fiches mémos. Il est assez difficile de faire ces fiches lorsque le chapitre devient plus grand ou plus dense en information. Rosalind est vraiment une bonne idée de mini projet pour débiter sur python.

J'ai trouvé l'échange de fiches à chaque CM, ennuyant à la fin. Le cours est bien structuré et compréhensible, j'ai trouvé du plaisir à le lire. Je retiens du positif dans ce cours, il m'a fait découvrir et aimer la programmation.

# Évaluation de l'enseignement (EEE)

Sans les fiches memo à faire je n'aurai jamais lu le poly correctement. Faire plus de script a trous ca nous force a réfléchir sur les variables à manipuler

présentiel. Selon moi le gros point fort est l'interactivité des cours, possible grâce à la pédagogie inversée, il faut absolument garder cette interactivité car c'est ce qui rend ce cours passionnant. Les activités en groupe de 2/3/4 sont aussi à garde car c'est une façon de nous faire travailler de manière ludique sans qu'on ai l'impression de travailler, et surtout cela donne l'envie d'aller plus loin dans la programmation. Pour ma part la



# Et alors ?

# Cette année

« Programmation Python 1 »



« Programmation Python **2** »

+ distanciel...



# Des nouveautés, distanciel-proof

Construction et évaluation de QCM par les étudiants

- Moodle : StudentQuiz

Projet « tuto » :

- Tutoriel vidéo sur une fonctionnalité non abordée en cours
- Travail en équipe
- Évaluation par les pairs



# Merci !

Me contacter :

- Mail : pierre.poulain@u-paris.fr
- Twitter : @pierrepo

Ce diaporama et plus encore :  
<https://cupnet.net/python-ci-2019>