

Programmation de Modèles Linguistiques 1, L6SOPRG L3

Caroline Koudoro-Parfait
caroline.parfait@sorbonne-universite.fr

Observatoire des Textes des Idées et des Corpus - Obtic,
Sorbonne Center for Artificial Intelligence - SCAI,
Sens Textes Informatiques Histoire - STIH EA 4509, Sorbonne Université

Organisation du semestre

CM + TD le Vendredi de 13h00 à 17h00 (C. Koudoro-Parfait & LG Moreno Jimenez)

Contrôle continu + projet + contrôle terminal

Ressources

TAL et Linguistique Informatique 1, ISTE Ed. (Mohamed Z. Kurdi)

Speech and Language Processing (Dan Jurafsky), <https://web.stanford.edu/~jurafsky/slp3/>

Helpdesk : mail ou bureau 206/211 à Serpente (sur RDV)

Convertir un jupyter notebook en script .py

Bonnes pratiques d'écriture d'un programme

Spyder un autre environnement

Listes, dico, set (et match !)

Convertir un jupyter notebook en script .py

Étapes importantes

- ✓ Commenter le code non essentiel
- ✓ Factoriser le code Jupyter Notebook en fonctions
- ✓ Créer des scripts Python pour chaque tâches associées

Étapes importantes

- ✓ Commenter le code non essentiel : #
- les parties de programme qui ne **fonctionnent pas** ou **expérimentatoires**.

Étapes importantes

- ✓ Commenter le code non essentiel : #
 - les parties de programme qui ne **fonctionnent pas** ou **expérimentatoires**.
- ✓ Factoriser le code Jupyter Notebook en fonctions. Observer votre notebook, n'y a t il pas :
 - des parties de code que vous répétez ?
 - des fonctions que vous répétez inutilement.

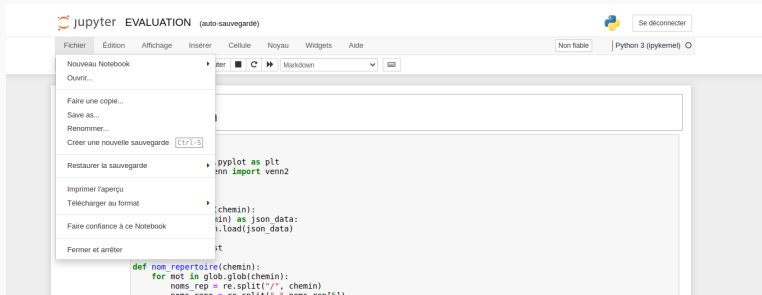
Étapes importantes

- ✓ Commenter le code non essentiel : #
 - les parties de programme qui ne **fonctionnent pas** ou **expérimentatoires**.
- ✓ Factoriser le code Jupyter Notebook en fonctions. Observer votre notebook, n'y a t il pas :
 - des parties de code que vous répétez ?
 - des fonctions que vous répétez inutilement.
- ✓ Créer des scripts Python pour des tâches associées.
 - Vous pouvez écrire un scripte initiale dans lequel vous appelez un autre script qui va contenir les fonctions que vous souhaitez utiliser

Enregistrer le fichier au format python - .py

Dans la barre de tâche en haut de l'écran jupyter notebook :

Aller dans Fichier → Télécharger au format → Python (.py)



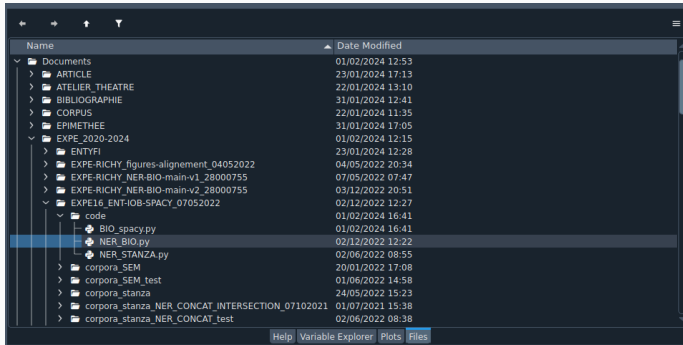
Bonne pratique d'écriture d'un programme

```
7  """
8  # _____ TOUS LES IMPORTS
9  import json
10 import glob
11 # _____ FIN DE TOUS LES IMPORTS
12
13 # _____ TOUTES LES FONCTIONS
14 def lire_fichier (chemin):
15     with open(chemin) as json_data:
16         dist =json.load(json_data)
17
18     return dist
19 # _____ FIN DE TOUTES LES FONCTIONS
20
21 # _____ MAIN
22 path_corpora = "./corpora/corpus_eval/*/*/"
23 # dans "corpora" un subcorpus = toutes les versions 'un texte'
24 liste_EN_ocr=[]
25 liste_EN_pp=[]
26 for subcorpus in sorted(glob.glob(path_corpora)):
27
28     for path in sorted(glob.glob("%s/*.json"%subcorpus)):
29         texte = lire_fichier(path)
30
```

- import
- fonctions
- Main :
 - appel des fonctions
 - boucles ...

Spyder un autre environnement

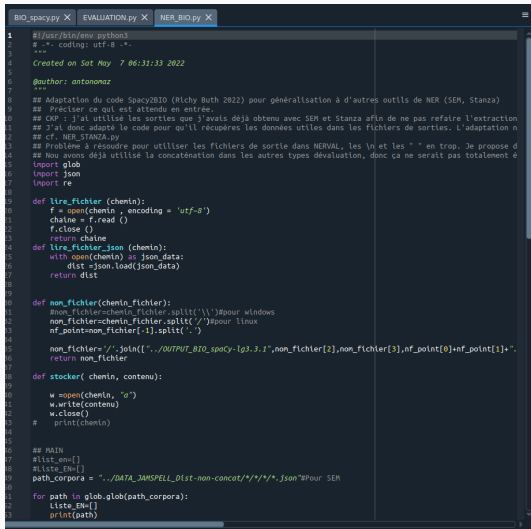
Sélectionner et ouvrir un fichier



Pour ouvrir un fichier vous pouvez :

- Sélectionner votre script dans l'explor. de fichier de votre machine et le déposer dans l'éditeur
- File → Open → Sélectionner votre script dans l'explor. de fichier affiché
- dans l'explorateur onglet → file → naviguer → double cliquer

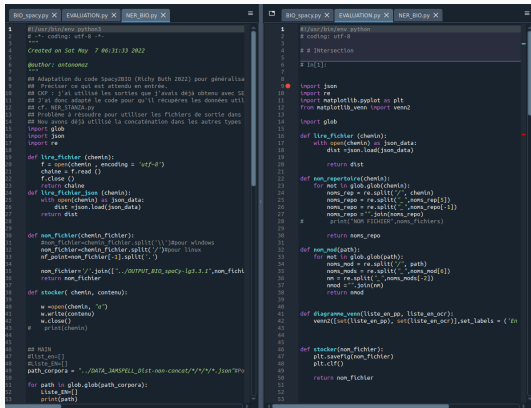
L'Éditeur 1



```
1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3 """
4 Created on Sat May 7 06:31:33 2022
5
6 @author: antonoz
7 """
8
9 ## Adaptation du code Spacy2Bio (Richy Buth 2022) pour généralisation à d'autres outils de NER (SEM, Stanza)
10 ## Préciser ce qui est attendu en entrée.
11 ## CKP : j'ai utilisé les sorties que j'avais déjà obtenu avec SEM et Stanza afin de ne pas refaire l'extraction
12 ## j'ai donc adapté le code pour qu'il récupère les données utiles dans les fichiers de sorties. L'adaptation n
13 ## est pas parfaite.
14 ## Problème à résoudre pour utiliser les fichiers de sortie dans NERVAL, les \n et les " en trop. Je propose d
15 ## Nous avons déjà utilisé la concaténation dans les autres types d'évaluation, donc ça ne serait pas totalement é
16
17 import glob
18 import json
19 import re
20
21 def lire_fichier(chemin):
22     f = open(chemin, encoding = 'utf-8')
23     chaine = f.read()
24     f.close()
25     return chaine
26
27 def lire_fichier_json(chemin):
28     with open(chemin) as json_data:
29         dist = json.load(json_data)
30         return dist
31
32 def nom_fichier(chemin_fichier):
33     #nom_fichier=chemin_fichier.split('\\')#pour windows
34     nom_fichier=chemin_fichier.split('/')#pour linux
35     nf_point=nom_fichier[-1].split('.')
36
37     nom_fichier='.'.join(['../OUTPUT_BIO_spacy-tp3.3.1',nom_fichier[2],nom_fichier[3],nf_point[0]+nf_point[1]+'-'])
38     return nom_fichier
39
40 def stocker(chemin, contenu):
41
42     w=open(chemin, "a")
43     w.write(contenu)
44     w.close()
45     # print(chemin)
46
47
48 ## MAIN
49 #liste_en=[]
50 #liste_EN=[]
51 path_corpora = "../DATA_JAMSPELL_Dist-non-concat/*/*/*.json"#Pour SEM
52
53 for path in glob.glob(path_corpora):
54     liste_EN=[]
55     print(path)
```

➔ +sieurs scripts peuvent être ouverts en même temps

L'Éditeur 2



```
1 #!/usr/bin/env python
2 # coding: utf-8
3 """
4 Created on Sat May 7 06:31:33 2022
5
6 @author: antonoz
7 """
8 ## Adaptation du code SpacyNER (Elchly Butz 2022) pour généraliser
9 ## Préciser ce qui est attendu en entrée.
10 ## On : j'ai utilisé les sorties que j'avais déjà obtenu avec SC
11 ## j'ai donc adapté le code pour qu'il récupère les données util
12 cf. NER_STANDA.py
13 ## Problème à résoudre pour utiliser les fichiers de sortie dans
14 ## New avons déjà utilisé la concaténation dans les autres types
15
16 import glob
17 import json
18
19 def lire_fichier(chemin):
20     f = open(chemin, encoding = 'utf-8')
21     chaine = f.read()
22     f.close()
23     return chaine
24
25 def lire_fichier_json(chemin):
26     with open(chemin) as json_data:
27         dist = json.load(json_data)
28     return dist
29
30 def nom_fichier(chemin_fichier):
31     nom_fichier = chemin_fichier.split("\\")[-1] pour windows
32     nom_fichier = chemin_fichier.split("/")[-1] pour linux
33     nf = nom_fichier.split("-")[-1].split(".")
34
35     nom_fichier = ".".join(["./OUTPUT_NER_nomCy-ldr.3.1", nom_fichier])
36     return nom_fichier
37
38 def stocker(chemin, contenu):
39
40     w = open(chemin, "a")
41     w.write(contenu)
42     w.close()
43     # print(chemin)
44
45
46 ## MAIN
47 # liste_em[]
48 # liste_em[]
49 path_corpora = "../DATA_SANSPELL_dist-non-concat/*/*/*.json"
50
51 for path in glob.glob(path_corpora):
52     liste_em[]
53     print(path)
```

```
1 #!/usr/bin/env python
2 # coding: utf-8
3
4 # Intersection
5
6 # la[]
7
8
9
10 import json
11 import re
12 import matplotlib.pyplot as plt
13 from matplotlib_venn import venn2
14
15 import glob
16
17 def lire_fichier(chemin):
18     with open(chemin) as json_data:
19         dist = json.load(json_data)
20
21     return dist
22
23 def nom_repertoire(chemin):
24     for mot in glob.glob(chemin):
25         noms_rep = re.split("/", chemin)
26         noms_repo = re.split("-", noms_rep[-1])
27         noms_repo = re.split("-", noms_repo[-1])
28         noms_repo = "".join(noms_repo)
29         print("NOM_FICHIER", noms_fichiers)
30
31     return noms_repo
32
33 def nom_mod(path):
34     for mot in glob.glob(path):
35         noms_mod = re.split("/", path)
36         noms_mods = re.split("-", noms_mod[-1])
37         nm = re.split("-", noms_mods[-1])
38         nomod = ".".join(nm)
39         return nomod
40
41 def diagramme_venn(liste_en_po, liste_en_ocr):
42     venn2([set(liste_en_po), set(liste_en_ocr)], set_labels = ('en
43
44
45
46 def stocker(nom_fichier):
47     plt.savefig(nom_fichier)
48     plt.clf()
49
50     return nom_fichier
51
52
53
```

➔ +sieurs scripts peuvent être ouverts côte à côte, ou l'un au dessus de l'autre.

➔ Cliquer sur



et sélectionner *split horizontally* ou *split vertically*

Situer son script sur sa machine



- Haut droite, fil d'Ariane : Indique où votre script est rangé sur la machine
- Haut gauche : permet de changer de dossier racine, d'explorer l'arborescence de votre machine

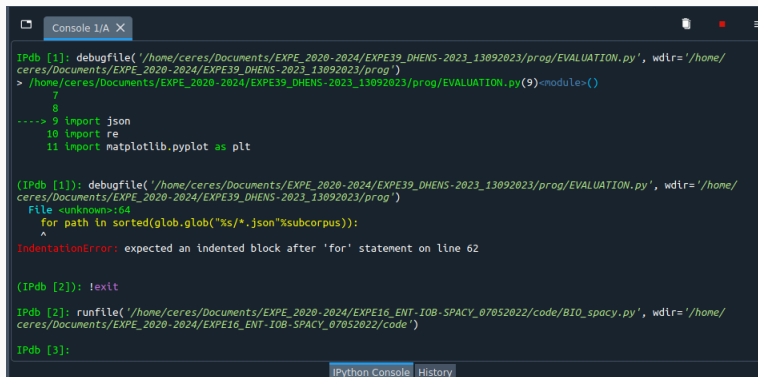
Renseignements sur l'environnement

conda: base (Python 3.11.5) Completions: conda(base) ✓ LSP: Python Line 24, Col 1 UTF-8 LF RW Mem 17%

- conda : base(3.11.5) → Version de python utilisé dans l'environnement
- Completions : conda(base) → Complétion automatique proposée
- LSP : Python → Language Server Protocol (LSP)
- Line 24, Col 1 → situation du curseur sur l'éditeur
- UTF-8 : Encodage
- LF → File EOL status, Linux : LF, Windows : CRLF. Problème de fin de ligne par exemple.
- RW → File permission ¹
- Mem 17% → Mémoire globale utilisée

1. <https://www.warp.dev/terminus/linux-file-permissions-explained>

La console



```
Console 1/A X

IPdb [1]: debugfile('/home/ceres/Documents/EXPE_2020-2024/EXPE39_DHENS-2023_13092023/prog/EVALUATION.py', wdir='/home/ceres/Documents/EXPE_2020-2024/EXPE39_DHENS-2023_13092023/prog')
> /home/ceres/Documents/EXPE_2020-2024/EXPE39_DHENS-2023_13092023/prog/EVALUATION.py(9)<module>()
7
8
----> 9 import json
      10 import re
      11 import matplotlib.pyplot as plt

(IPdb [1]): debugfile('/home/ceres/Documents/EXPE_2020-2024/EXPE39_DHENS-2023_13092023/prog/EVALUATION.py', wdir='/home/ceres/Documents/EXPE_2020-2024/EXPE39_DHENS-2023_13092023/prog')
File <unknown>:64
    for path in sorted(glob.glob("%s/*.json"%subcorpus)):
    ^
IndentationError: expected an indented block after 'for' statement on line 62

(IPdb [2]): !exit

IPdb [2]: runfile('/home/ceres/Documents/EXPE_2020-2024/EXPE16_ENT-IOB-SPACY_07052022/code/BIO_spacy.py', wdir='/home/ceres/Documents/EXPE_2020-2024/EXPE16_ENT-IOB-SPACY_07052022/code')

IPdb [3]:

IPython Console History
```

- des lignes de codes peuvent être lancées depuis la console
- Les messages d'erreurs s'affichent dans la console → Débogage
- Carré rouge en haut à droite de la console == ordinateur calcul

Débogage base

```
60 liste_EN_pp=[]
61 #for subcorpus in sorted(glob.glob("%s/*"%path_corpora)):
62 for subcorpus in sorted(glob.glob(path_corpora)):
63     # print(subcorpus)
64     for path in sorted(glob.glob("%s/*.json"%subcorpus)):
65         # print("subsubcorpus",path )
66
67         texte = lire_fichier(path)
68         # print("*****",subcorpus)
69         # print(texte)
70         nomrep=nom_repertoire(path)
71         # print(nomrep)
72
```

→ affiche qu'il y a un problème, passer la souris / cliquer dessus pour avoir le détail.

Historique

```
history.py

# -*- coding: utf-8 -*-
# *** Spyder Python Console History Log ***

## ---(Thu Feb 1 12:29:55 2024)---
runfile('/home/ceres/.config/spyder-py3/temp.py', wdir='/home/ceres/.config/spyder-py3')

## ---(Thu Feb 1 12:38:55 2024)---
runfile('/home/ceres/.config/spyder-py3/temp.py', wdir='/home/ceres/.config/spyder-py3')

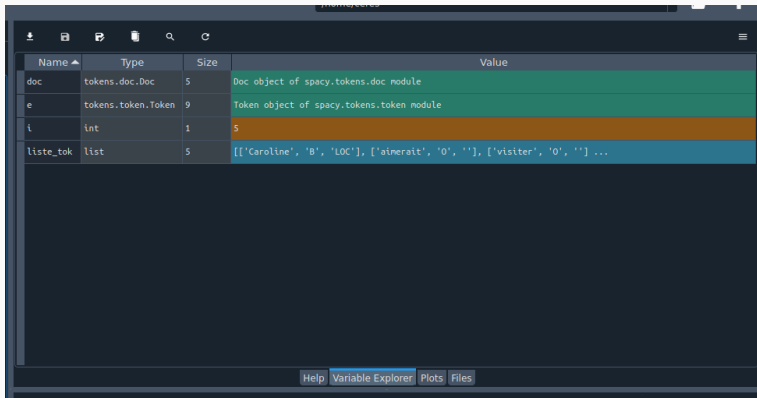
## ---(Thu Feb 1 12:50:47 2024)---
runfile('/home/ceres/.config/spyder-py3/temp.py', wdir='/home/ceres/.config/spyder-py3')

## ---(Thu Feb 1 16:25:17 2024)---
runfile('/home/ceres/.config/spyder-py3/temp.py', wdir='/home/ceres/.config/spyder-py3')
runfile('/home/ceres/Documents/EXPE_2020-2024/EXPE16_ENT-IOB-SPACY_07052022/code/BIO_spacy.py', wdir='/home/ceres/Documents/EXPE_2020-2024/EXPE16_ENT-IOB-SPACY_07052022/code')

## ---(Thu Feb 1 18:05:11 2024)---
debugfile('/home/ceres/Documents/EXPE_2020-2024/EXPE39_DHENS-2023_13092023/prog/EVALUATION.py', wdir='/home/ceres/Documents/EXPE_2020-2024/EXPE39_DHENS-2023_13092023/prog')
runfile('/home/ceres/Documents/EXPE_2020-2024/EXPE16_ENT-IOB-SPACY_07052022/code/BIO_spacy.py', wdir='/home/ceres/Documents/EXPE_2020-2024/EXPE16_ENT-IOB-SPACY_07052022/code')
for e in doc:
```

➔ Consulter l'historique des commandes

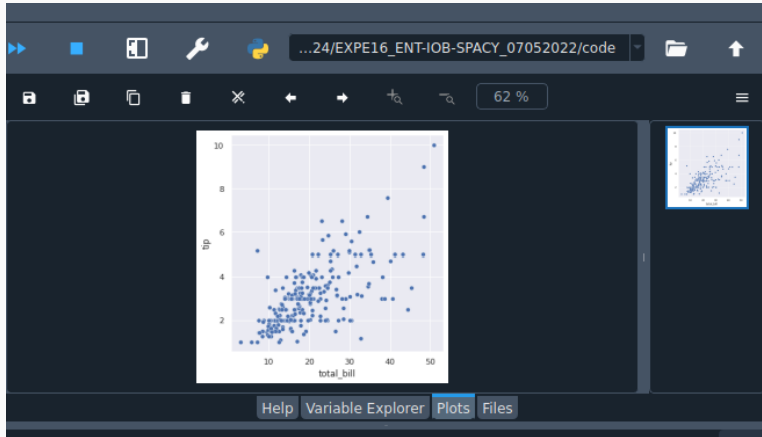
L'explorateur de variable



Name ▲	Type	Size	Value
doc	tokens.doc.Doc	5	Doc object of spacy.tokens.doc module
e	tokens.token.Token	9	Token object of spacy.tokens.token module
i	int	1	5
liste_tok	list	5	[['Caroline', 'B', 'LOC'], ['aimerait', 'O', ''], ['visiter', 'O', '']] ...

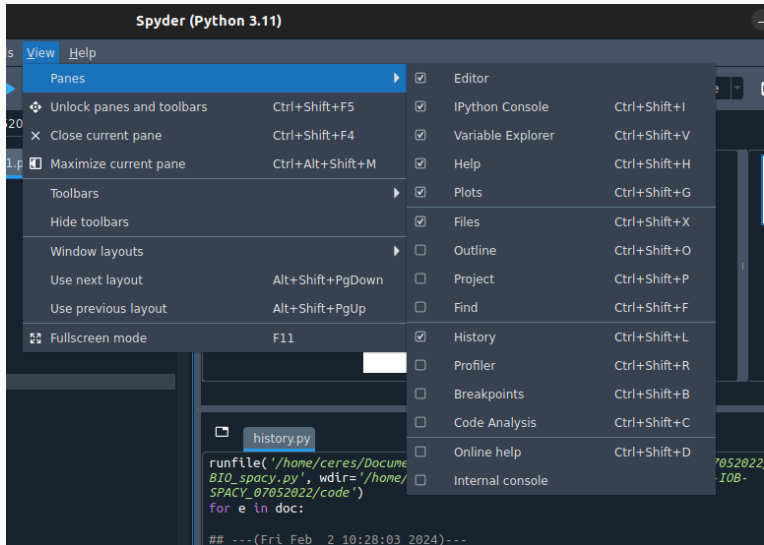
→ Accédez au nom de variable, type, longueur du contenu de la variable et valeurs

Affichage des graphiques - plots



→ L'affichage des graphiques ce fait dans ce panneau.

Affichage des panneaux (explo. variable, console...)



→ Paramétrer l'affichage des panneaux que vous souhaitez voir affichés ou non.

Listes, dico, set (et match !)

Petit point sur les listes

➔ Une liste est une structure de données permettant d'accéder à un objet par son index.²

➔ Pour illustrer :

☆ supposons qu'une liste représente une rue.

☆ Chaque habitant vit dans une maison avec un numéro.

☆ Ce numéro est l'index de la liste grâce auquel on peut accéder à l'habitant .

Syntaxe :

* `liste = []` :définition d'une liste vide

* `liste = ["Annie" , "Paul"]` :défnition avec quelques éléments

* `liste [0]` vaut "Annie" , `liste[1]` vaut "Paul" .

2. + d'infos sur les listes <https://gayerie.dev/docs/python/python3/list.html>

Petit point sur les dictionnaires

- Un dictionnaire est une structure de donnée permettant d'accéder à un objet par une clef.³
 - Dans un dictionnaire de langue, on utilise les mots comme clef afin d'accéder à leur définition, qui sont ici les objets.
 - Dans un dictionnaire, chaque **clef est unique**.
 - Par contre, **une même définition** peut correspondre à **+sieurs clef**.
- Syntaxe :

```
* dico_vide = {}  
* dico_age = {"Annie" : 20, "Paul" : 18, "Antonia" : 20}  
* dico_age [ "Annie" ] = 20 .
```

3. + d'infos sur les dict. <https://gayerie.dev/docs/python/python3/dict.html>

Petit point sur les set - ensembles

- c'est un groupement non ordonné d'éléments uniques → pas de doublon⁴.
- peut contenir des éléments de types différents.
- pas possible d'accéder aux éléments d'un ensemble avec l'opérateur []

Syntaxe :

```
* mon_ensemble_vide = set() != mon_dico = {}  
* mon_ensemble = {None, 10, "Bonjour", True}  
* 10 in mon_ensemble → True  
* 12 in mon_ensemble → False  
* 12 not in mon_ensemble → True
```

4. + d'infos sur les set <https://gayerie.dev/docs/python/python3/set.html>

Petit point sur n-uplet ou tuple

- Séquence non modifiable de données ordonnées⁵.
- Chaque élément du tuple est associé à une position (un index).
- Les tuples ne sont pas modifiables.

Syntaxe :

```
* tuple_vide = ()  
* mon_tuple = (10, 20, 30, 40)  
* mon_tuple = ("caroline",) != chaine="caroline"  
* tuple_depareille = (None, 10, "Bonjour", True)
```

5. + d'infos sur les tuple <https://gayerie.dev/docs/python/python3/tuple.html>

- Les ensembles, listes et tuples sont des collections d'éléments
- il est possible de passer de l'un à l'autre grâce aux **méthode** `set()`, `list()`, `tuple()`.
- un ensemble/set n'est **pas ordonné** != liste, un tuple ordonné
- parcourir un set avec une instruction `for`, pas de garantie sur l'ordre de parcours des éléments.
- un set n'est pas une structure de données pertinente quand il existe une relation d'ordre entre les éléments.