École nationale de la statistique et de l'analyse de l'information

Python pour l'ingénieur
2A 2016-2017

TP 1 : Bases du langage de programmation Python

Ce TP à pour objectif de vous faire découvrir le langage de programmation Python à l'aide de l'IDE Spyder et de l'interpréteur interactif ipython (console Spyder).

1 Découverte de l'IDE : Spyder

Spyder est un environnement de développement (Integrated Development Environment) pour Python, fourni avec toutes les versions de python.

L'utilisation de Spyder est très simple, et l'interface comporte par défaut une fenêtre avec ces panneaux :

- Sur la gauche, un éditeur de texte, dans lequel vous pouvez entrer votre programme (avec un code couleur et indentation automatique), l'enregistrer et l'exécuter ;
- En bas à droite, un shell (ou interprète interactif) permettant de visualiser l'exécution du programme, et de lancer le débogueur ;
- En haut à droite, une fenêtre permettant d'afficher des informations sur le programme.

1.1 L'interprète Python

1.1.1 Exercice 1 (Evaluation de quelques expressions)

Évaluez les expressions suivantes dans l'interprète :

- 2 + 2
- 2 - 2
- -2
- 2 + 3 * 4
- (2 + 3) * 4
- 2 ** 3
- 2 ** 4
- 10 / 3
- 10 // 3
- 10 % 3

1.1.2 Exercice 2 : Les chaînes de caractères

Une chaîne de caractères s'écrit en plaçant cette chaîne soit entre deux guillements (ex. "Bonjour") soit entre deux simple quotes (ex. 'Bonjour').

1. Affectez la chaîne de caractères "salut" à la variable s.
2. Tapez l'instruction `s?`. À quoi cela correspond-il ?

3. Il est possible de faire de l'autocomplétion avec ipython. Taper `s` suivi d'une tabulation. Que se passe-t-il ?

4. Tapez ensuite `s.` suivi d'une tabulation pour obtenir la liste des différents champs et méthodes de la classe string. Vous noterez que les méthodes préfixées et suffixées par `__` ne sont généralement pas utilisées telles quelles. Il existe souvent des alternatives : par exemple, $s.\_\_len\_\_() \Longleftrightarrow len(s)$.

5. Pour obtenir la documentation d'une méthode, taper le nom de la méthode suivi du symbole '?' (ex. s.replace?). Tester l'exemple. Que fait cette méthode ? Utilisez cette méthode.

6. Trouvez un moyen de mettre en majuscule la première lettre de s.

7. Testez les opérateurs (+,-,*) sur deux chaînes de caractères.

8. Testez ces mêmes opérateurs sur une chaîne de caractères et d'autres types (ex. entier).
9. Testez aussi les comparateurs.

1.2 L'éditeur Python

1.2.1 Exercice 3 : Les scripts

Les instructions Python peuvent être entrées directement dans l'interpréteur, mais ce n'est évidemment pas pratique dans le cas d'un programme complet.

Pour exécuter plusieurs instructions les unes après les autres (i.e un programme), elles peuvent être écrites dans un fichier (aussi appelé script), nommé par exemple : exercice1.py (l'extension .py est vivement recommandée). Ce script pourra alors être exécuté avec la commande `%run exercice1.py` dans un console.

1. Écrire un script définissant une liste contenant 5 éléments et affichant ces derniers un par un.

2. Écrire un script permettant de comparer la longueur de deux mots et d'afficher le plus long.

3. Écrire un script dans lequel vous définirez une liste d'entiers allant de 1 à 20 (utilisez la fonction `range`),

4. Utilisez une boucle et des conditions pour n'afficher que les nombres pairs.

5. Essayez d'ajouter une erreur dans votre script, par exemple remplacer print(a) par print(a (suppression de la parenthèse fermante). Enregistrez le fichier.
   Spyder vous signale une erreur avec un symbole ⚠ dans la marge à côté du **print**.
   On peut tout de même tenter une exécution : Spyder vous signale alors une erreur de syntaxe dans le panneau « Console ». En cliquant sur le message d'erreur, le curseur se positionne à la bonne ligne du fichier concerné. Corrigez l'erreur (la parenthèse fermante manquante du print) et exécutez le programme.

Exercice 4 : Plus de pratique

Réaliser des programmes qui répondent aux questions suivantes :

1. **Somme des n premiers entiers.** Calculer $\sum_{i=1}^{n} i$.

2. **Énumérer les diviseurs d'un nombre entier.** Écrire une fonction qui retourne la liste des diviseurs d'un nombre entiers.

3. Calculer $\sum_{i=1}^{10} \frac{x^{2\,i}}{sin\, i}$.

4. **Factorielle**. Écrire une fonction récursive qui calcule la factorielle de n.

5. Écrire une fonction qui vérifie qu'une chaîne de caractères est symétrique (Palindrome). Exemple : kayak, a man a plan a canal panama.

6. Écrire une fonction qui prend une chaîne de caractères en argument et

- a) retourne la chaîne sans ses voyelles.
- b) met en majuscule les voyelles. (Proposez 2 versions)
- c) mélange aléatoirement les lettres de la chaîne de caractères.

7. On considère une liste de mots :

```
mots = ['eddard', 'catelyn', 'robb', 'sansa', 'arya', 'brandon',
        'rickon', 'theon', 'rorbert', 'cersei', 'tywin', 'jaime',
        'tyrion', 'shae', 'bronn', 'lancel', 'joffrey', 'sandor',
        'varys', 'renly', 'a' ]  \medskip
```

   - Écrire une fonction qui retourne tous les mots de la liste qui ont un 'y' en seconde position.

```
def mots_lettre_position (liste, lettre, position) :
    # ......
    return [ .... ]
```

   - Écrire une fonction qui compte le nombre de chaînes de caractères où la taille du mot est supérieure ou égale à deux et le premier et le dernier caractère est identique.

8. Compter le nombre d'occurrences de chaque caractère dans la chaîne de caractères "HelLo WorLd!!"

9. Transformer une matrice représentée sous forme de double liste (exemple : [[0,1,0],[0,0,1]]) en un dictionnaire dont les clés sont les coordonnées et les valeurs les coefficients (soit autant d'éléments que de valeurs non nulles). *Indice :* Pensez à utiliser la fonction `enumerate`.

10. **Retourner un dictionnaire :** Donnez un programme qui permet de retourner un dictionnaire, c'est-à-dire : les clés deviennent les valeurs et les valeurs deviennent les clés (on suppose que les clés et les valeurs sont uniques).

11. **Suite de Fibonacci**. La suite de Fibonacci est définie par la relation de récurrence suivante :

$$\begin{cases} u_0 = 0 \\ u_1 = 1 \\ u_n = u_{n-2} + u_{n-1} \quad \text{pour } n \geq 2 \end{cases}$$

   - Écrivez une fonction Fibonacci(un_2, un_1) qui calcule le terme de la suite de Fibonacci qui vient juste après un_1, un_2.
   - Écrivez une fonction liste_fibonacci(n) qui retourne une liste des n premiers termes de la suite de Fibonacci.
   - Calculez maintenant la séquence des rapports u_{n+1}/u_n et vérifiez qu'elle converge vers le nombre d'or $\phi = \frac{1+\sqrt{5}}{2}$. Si on note l la liste des termes de la suite, on pourra générer la liste des rapports par compréhension en parcourant en parallèle les deux sous-listes l[:-1], l[1 :] en utilisant la fonction zip.

```
In[] : l = ["a", "b", "c"]
In[] : [(u1, u2) for (u1, u2) in zip(l[:-1], l[1:])]
[ ("a","b"), ("b", "c") ]
```

Exercice 5 : Deviner un nombre aléatoire

1. Définir une fonction qui demande à l'utilisateur de saisir un nombre.
2. Écrire un jeu dans lequel python choisi aléatoirement un nombre entre 0 et 100, et essayez de trouver ce nombre en 10 étapes.
3. Transformer ce jeu en une fonction jeu(nVies) où nVies est le nombre d'itérations maximum.

Exercice 6 : Trouver l'erreur

1. Le programme suivant provoque une erreur de temps en temps, pourquoi ?

```
l = [0, 1, 2, 3, 4]
i = random.randint(0, 5)
del l[i] # déclenche une exception de temps en temps
```

2. A votre avis, qu'a voulu dire l'auteur de ces lignes ? Corriger ce programme.

```
mat = {}
for i in range(0,3) and j in range(0,3): # déclenche une exception
    mat[i,j] = 0
```

3. Téléchargez le fichier TrouvezErreur.py sur moodle. Enregistrez le fichier et ouvrez-le sous Spyder. Trouvez et corrigez toutes les erreurs dans les fonctions Python données.

Exercice 7

On s'intéresse à une phase du jeu 2048. On part d'une grille :

```
mat = [[2,0,0,4],[0,2,8,2],[0,2,4,2],[2,2,8,0],]
for m in mat: print(m)
```

```
[2, 0, 0, 4]
[0, 2, 8, 2]
[0, 2, 4, 2]
[2, 2, 8, 0]
```

1. On veut écrire une fonction qui calcule l'état du jeu après la pression de la touche **bas**.

```

```

2. Modifier la fonction pour gérer les quatre directions.

```

```

3. Remplir deux cases vides choisies aléatoirement avec deux chiffres 2.

```

```