Haute École Bruxelles-Brabant
École Supérieure d'Informatique
Bachelor en Informatique

2025 – 2026
1DEV1A – 1DEVL1A

1DEV1A – 1DEVL1A – Laboratoires Python

TD 08 – Boucles For

Nous allons maintenant entamer notre initiation à la rédaction de boucles `for` en Python. Ces boucles auront pour objectif de faire exécuter un groupe d'instructions un certain nombre de fois selon une valeur préalablement déterminée.

Table des matières

Version du document
7 octobre 2025

Rappels

- Vous devez absolument avoir fait et compris les TD précédents *avant* d'aborder celui-ci.
- Comme indiqué dans le TD05, soyez attentif, dans ce TD et des suivants, à *toujours* découper votre code en modules et programme principal. Nous ne le précisons pas à chaque exercice, mais la demande est implicite.

1 Boucle for avec une liste

Bien que le principe reste toujours le même, il existe plusieurs façons d'écrire une boucle `for` en Python. Nous allons en montrer quelques-unes ici, ainsi que certaines fonctions très utiles pour les boucles `for`.

Le script Python ci-dessus affiche les nombres de 1 à 5 au moyen d'une boucle `for`.

forList.py
```
1 for i in [1, 2, 3, 4, 5]:
2     print(i)
```

Une boucle `for` contient plusieurs éléments.

Comme une boucle `while`, elle contient un *entête* qui se termine par `:`.

Cet entête est suivi d'un *bloc d'instructions*, ici `print(i)`.

L'entête contient trois éléments. D'abord une variable, que l'on a nommé ici `i`, qui est aussi appelée la *variable de boucle*, ensuite un mot-clé `in` et, finalement, une liste `[1, 2, 3, 4, 5]`. Notez que la liste ne doit pas être obligatoirement une liste de nombres entiers.

Nous avons déjà vu ce qu'est une liste et quelques manipulations de base au premier TD et dans certains suivants. Ces connaissances sont suffisantes pour ce TD. Dans un prochain TD, nous étudierons les listes plus en détail.

À chaque fois que le bloc d'instruction est exécuté, on dit que le programme a fait une *itération* de la boucle `for`. À chaque itération, la variable que l'on a déclarée dans l'entête, la variable de boucle, va prendre successivement les valeurs qui se trouvent dans la liste. Une fois que la variable de boucle a sa valeur attribuée, le programme va exécuter le bloc d'instruction.

Exercice 1 Boucles for

Modifiez la liste dans la boucle ci-dessus de façon à ce que celui-ci :

1. affiche les entiers impairs de 1 à 9 ;
2. affiche les nombres 0, 0.25, 0.5, 0.75 et 1 ;
3. affiche les mots `une`, `boucle`, `for`.

2 Boucle for avec la fonction range

Quand on utilise des boucles, c'est principalement pour ne pas devoir écrire à la main toutes les valeurs des itérations. Or, ici, en écrivant notre liste, nous écrivons toutes les valeurs que la variable de boucle va prendre. Il existe en Python une fonction qui permet de remédier à ce problème.

2.1 Fonction range

Le script suivant permet d'afficher les nombres de 0 à 99.

forRange.py
```
1 for i in range(0, 100):
2     print(i)
```

La fonction `range`, permet de créer une séquence d'entiers compris entre les deux nombres passés en argument, le premier inclus, le second non. Ainsi dans l'exemple la fonction `range` va de 0 à 100, mais elle génère les nombres de 0 à 99.

Notez que si un seul argument est écrit dans la fonction `range`, alors la fonction fera une séquence d'entier qui commence à 0 et fini à cet argument. Par exemple `range(10)` crée une séquence qui va de 0 à 9, inclus.

On peut ajouter un troisième argument à cette fonction `range`. Celui-ci détermine le *pas* entre chaque valeur de la séquence. Par exemple, `range(0, 10, 2)` va générer la séquence `0, 2, 4, 6, 8`.

Exercice 2 Fonction range

Modifier le script pour qu'il demande à l'utilisateur un nombre entier n, le script doit alors

- ▷ afficher tous les nombres entiers entre 1 et n ;
- ▷ afficher tous les nombres impairs entre 1 et n, sans utiliser d'alternative ;
- ▷ afficher tous les nombres entre $-n$ et n.

3 Boucle for avec du texte

Plutôt que mettre une liste (ou un `range`) dans la boucle `for`, on peut également mettre du texte.

forChaine.py
```
1 for i in "esi":
2     print(i)
```

La variable de boucle prend alors la valeur de chaque caractère de cette chaîne de texte.

Exercice 3 Texte

Modifier le script pour qu'il demande à l'utilisateur un mot. Le script doit afficher chaque lettre de ce mot.

4 Boucle for et la fonction enumerate

Il est parfois utile d'avoir, en plus de la variable de boucle qui parcourt une liste (ou la fonction range ou du texte), une deuxième variable de boucle qui affiche le nombre d'itérations. Il est possible de faire cela avec vos connaissances actuelles.

Exercice 4 Compteur itération

Écrivez un programme qui demande à l'utilisateur un mot. Pour toutes les lettres dans ce mot le programme affiche la place de la lettre dans le mot ainsi que cette lettre.

Par exemple, si le mot est « for », le programme devra afficher :

```
La lettre 1 est un f
La lettre 2 est un o
La lettre 3 est un r
```

4.1 Fonction enumerate

En Python, la fonction `enumerate` nous permet de nous faciliter la vie.

Le code suivant va nous permettre de répondre à l'exercice précédent pour le mot « boucle ».

forEnumerate.py
```
1 for i, j in enumerate("boucle"):
2     print("la lettre", i, "est un", j)
```

Remarquer que, dans ce cas, il faut déclarer *deux* variables de boucle dans la boucle `for`. La première va compter le nombre d'itération de la boucle, la seconde va parcourir le mot (ou la liste etc.).

Exercice 5 Fonction enumerate

Pour chaque nombre de la liste [76, 23, 18, 6, 5, 2], calculez et affichez ce nombre à la puissance égale à sa place dans la liste.

Par exemple, pour le premier nombre, il faut calculer 76^1, pour le deuxième, il faut calculer 23^2, et ainsi de suite.

Aide : utilisez la fonction `enumerate`.

5 Exercices récapitulatifs

Exercice 6 La somme de carrés

Écrivez un programme qui fait la somme des carrés des 100 premiers entiers.

Exercice 7 Consonne ou voyelle

Écrivez un programme qui demande à l'utilisateur un mot. Pour chaque lettre de ce mot, le programme dit s'il s'agit d'une consonne ou d'une voyelle.

Exercice 8 Bagages dans l'avion

Écrivez un programme qui demande à l'utilisateur un nombre entier entre 1 et 10. Ce nombre représente le nombre de passagers dans un avion.

Pour chaque personne, le programme demande son nom.

Le programme demande alors, pour chaque passager, le poids de sa valise.

Pour chaque personne qui a une valise d'un poids supérieur à 10 kg, le programme affiche le nom de la personne et lui signale qu'il devra payer un supplément.

Le programme calcule finalement le poids total de toutes les valises et l'affiche.

Exercice 9 Palindrome

Un *palindrome* est un (ensemble de) mot(s) qui peu(ven)t aussi bien se lire de gauche à droite que de droite à gauche, en gardant la même signification [1]. Le mot « kayak » en est un exemple.

1. Si on lève la contrainte de préservation de la signification, on parle d'un *anacyclique*, comme dans « ados » et « soda » ou dans « super star » et « rats repus », par exemple.

Écrivez un programme qui demande à l'utilisateur un mot. Une fois vérifié qu'il s'agit bien d'un mot, le programme doit dire si la première lettre est la même que la dernière, si la deuxième est la même que l'avant-dernière et ainsi de suite. À la fin, le programme doit alors dire si le mot est un palindrome ou non.

Aide : soit `txt` une variable qui contient du texte, vous pouvez inverser ce texte via les instructions `txtInverse = txt[::-1]`.

6 En résumé ...

Principaux points de matière du TD

Voici les principaux points abordés lors de ce TD. Vous devez absolument être à l'aise avec ceux-ci avant d'aborder la prochaine séance d'exercice.

1. L'instruction `for` en Python.
2. L'utilisation des fonctions `range` et `enumerate` dans les boucle `for`.
3. Mettre en pratique la boucle `for` pour écrire des scripts résolvants des problèmes divers.