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

2025 – 2026
1DEV1A – 1DEVL1A

1DEV1A – 1DEVL1A – Laboratoires Python

TD 07 – Boucles While

Nous allons à présent nous initier à l'écriture de boucles `while` en Python de façon à exécuter un bloc de code tant qu'une condition est satisfaite.

Table des matières

Version du document
2 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 Boucles while

Le script Python ci-dessus affiche les nombres de 1 à 100 au moyen d'une boucle `while`.

while.py

```
compteur = 1

while compteur <= 100:
    print(compteur)
    compteur = compteur + 1

print("Terminé !")
```

L'instruction `while` permet d'exécuter un bloc de code tant qu'une condition, celle qui suit le mot clé `while`, est satisfaite, c'est-à-dire `True`.

Notez que le corps de la boucle modifie l'état du programme. Dans ce cas-ci, il s'agit de la variable `compteur`, initialisée préalablement. Cela a pour conséquence que la condition finit par être fausse, et permet au programme de s'arrêter.

Exercice 1 Boucles while

Modifiez le script ci-dessus de façon à ce que celui-ci :

1. demande un nombre entier n à l'utilisateur ;
2. affiche à l'écran tous les nombres entiers de 1 à n, où vous pouvez supposer que $n \geqslant 1$.

Exercice 2 Boucles while dans une fonction

Modifiez le script ci-dessus de façon à en faire deux fonctions :

- ▷ l'une s'occupe de demander un entier à l'utilisateur ;
- ▷ l'autre s'occupe d'afficher les entiers de 1 à n à l'écran.

Voici la structure que prendra votre fichier :

while_fonction

```
def demander_entier():
    # votre code

def afficher_premiers_entiers(n):
    # votre code

# l'appel principal :
afficher_premiers_entiers(demander_entier())
```

Écrivez des fonctions

Dans la suite, songez à toujours écrire une *fonction* qui répond à l'exercice, et à appeler cette fonction dans votre code principal.

Exercice 3 Suites d'entiers

Écrivez un script Python qui demande à l'utilisateur un nombre entier n et affiche :

- ▷ les nombres pairs qui sont compris entre 1 et n ;
- ▷ les nombres de $-n$ à n ;
- ▷ les multiples de 5 qui sont compris entre 1 et n ;
- ▷ les multiples de n compris entre 1 et 100.

2 Tester les entrées de l'utilisateur

Une utilisation classique de la boucle `while` est d'exécuter un morceau de code tant que les données entrées par l'utilisateur ne satisfont pas une certaine condition. Voici un exemple : le script suivant calcule la racine carrée d'un nombre entré par l'utilisateur, en s'assurant auparavant que le nombre entré est bien positif.

nombrepositif.py

```
1  import math
2
3  nombre = int(input("Veuillez entrer un nombre positif : "))
4
5  while nombre < 0:
6      nombre = int(input("Ce nombre n'est pas postif. Veuillez réessayer : "))
7
8  print("La racine carrée de", nombre, "est", math.sqrt(nombre))
```

Exercice 4 Le nom du mois

Écrivez un script Python qui demande à l'utilisateur un nombre entier compris entre 1 et 12 et affiche le nom du mois correspondant : 1 = «Janvier», 2 = «Février»..., 12 = «Décembre». Tant que le nombre entré n'est pas compris entre 1 et 12, le programme affiche un message d'erreur et en demande un autre à l'utilisateur.

2.1 Petit jeu

Voici un exemple de petit jeu : on demande à l'utilisateur de deviner un nombre secret entre 1 et 10 et on boucle tant que l'utilisateur ne l'a pas trouvé.

devinette.py

```
1  nombre_secret = 6
2
3  print("L'ordinateur a choisi un nombre entre 1 et 10...")
4  print("  Vous allez devoir le deviner !")
5
6  nombre_utilisateur = int(input("Entrez un nombre :"))
7
8  while nombre_utilisateur != nombre_secret:
9      nombre_utilisateur = int(input("Raté ! Essayez encore : "))
10
11 print("Bravo !")
```

Évidemment, le script ci-dessus a peu d'intérêt vu que le nombre à deviner est toujours égal à 6. À vous de le rendre plus intéressant.

Exercice 5 Le nombre secret

Modifiez le script ci-dessus de façon à ce que l'ordinateur choisisse réellement un nombre à deviner au hasard entre 1 et 10.

Rappel : nous avons vu dans le TD03 comment générer des nombres au hasard.

Exercice 6 Le nombre secret (suite)

Modifiez votre script sur le nombre secret de façon à compter et afficher le nombre d'essais que l'utilisateur a dû faire pour réussir à trouver le nombre. Voici un exemple d'exécution possible du script où l'utilisateur trouve le nombre en 5 essais :

```
L'ordinateur a choisi un nombre entre 1 et 10...
  Vous allez devoir le deviner !
Entrez un nombre: 1
Raté ! Essayez encore : 9
Raté ! Essayez encore : 5
Raté ! Essayez encore : 7
Raté ! Essayez encore : 8
Bravo ! Vous avez trouvé le nombre secret en 5 essais.
```

2.2 Valeur sentinelle

Voici encore un exemple de boucle dont la condition dépend des entrées au clavier. On lit des nombres au clavier tant qu'ils sont positifs, ou, dit autrement, tant que l'utilisateur ne rentre pas une valeur négative.

Concrètement, le nombre -1 est utilisé comme *valeur sentinelle* : on continue de boucler tant que le programme ne détecte pas de valeur inférieure ou égale à -1, ou, plus simplement, on boucle tant que des valeurs strictement supérieures à -1 sont lues.

Le corps de la boucle peut alors traiter le nombre entré au clavier. Le traitement dépend de ce qu'on souhaite faire de la suite de nombres en question.

suitenombre.py

```
# Canevas d'un script lisant une suite de nombres au clavier
nb_valeurs = 0

valeur = int(input("Entrez une suite de nombre entier positifs, terminez par -1 : "))

while valeur > -1:
    nb_valeurs = nb_valeurs +1
    # Faire quelque chose avec le contenu de la variable "valeur" ici
    valeur = int(input("Entrez une autre valeur (ou -1 pour terminer) : "))

print("Vous avez entré", nb_valeurs, "valeur(s)")
```

Exercice 7 La somme

Adaptez le canevas ci-dessus de façon à lire une suite de nombre terminée par -1 et à afficher la somme à l'écran.

Par exemple, si l'utilisateur entre : `2 6 3 1 2 -1`, le script affiche `La somme vaut 14`.

Exercice 8 La moyenne

Modifiez le script précédent de façon à ce que le programme affiche la moyenne, et non plus la somme, de tous les nombres entrés. De plus, si l'utilisateur ne rentre aucun nombre positif, le programme affiche un message d'erreur.

3 Exercices récapitulatifs

Exercice 9 Les nombres carrés

Écrivez un script qui demande un nombre positif n à l'utilisateur et qui affiche tous les nombres carrés [1] plus petits ou égaux à n.

1. https://fr.wikipedia.org/wiki/Carr%C3%A9_parfait

Par exemple, si l'utilisateur entre 30, le script affiche `1 4 9 16 25`, c'est-à-dire les carrés de 1, 2, 3, 4 et 5, respectivement.

Exercice 10 Le mot

Écrivez un script qui demande un mot à l'utilisateur. Un mot est une chaîne de caractère ne contenant aucun espace. Tant que la chaîne entrée par l'utilisateur contient un espace, le script affiche un message d'erreur et demande un autre mot à l'utilisateur.

Aide : relisez le TD4 pour savoir comment tester qu'une chaîne contient un espace.

Exercice 11 Le nombre secret (suite et fin)

Modifiez à nouveau votre script sur le nombre secret de façon à aider le joueur. En fonction du nombre erroné entré par l'utilisateur, le programme afficher `Trop petit` ou `Trop grand`.

Voici un exemple d'exécution possible du script :

```
L'ordinateur a choisi un nombre entre 1 et 10...
  Vous allez devoir le deviner !
Entrez un nombre: 1
Trop petit ! Essayez encore : 9
Trop grand ! Essayez encore : 6
Trop grand ! Essayez encore : 5
Bravo ! Vous avez trouvé le nombre secret en 4 essais.
```

Exercice 12 Maximum et minimum

Écrivez un script qui demande à l'utilisateur une série de nombres positifs qui se termine par -1, valeur sentinelle. Le programme affiche le maximum et le minimum de la série.

Par exemple, si l'utilisateur entre : 5 9 2 5 12 7 4 3 -1, le programme affiche :

```
maximum : 12
minimum : 2
```

N'oubliez pas de bien prendre en compte le cas où l'utilisateur ne fournit aucune valeur positive.

4 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 `while` en Python.
2. L'utilisation de la boucle `while` pour vérifier les entrées de l'utilisateur.
3. Mettre en pratique la boucle `while` pour écrire des scripts résolvants des problèmes divers.