□ La Boucle while

Objectif : Savoir utiliser la boucle **while** pour répéter un bloc tant qu'une condition est vraie. À la différence de **for**, on utilise **while** quand on **ne sait pas à l'avance** combien de fois on va répéter.

□ Concept

python

```
while condition :
    # traitement
    # modifier la condition ! ← sinon boucle infinie
```

Opération	Code Python
Répéter tant que condition vraie	`while condition:`
Répéter jusqu'à trouver	`while not condition:`
Valider une saisie	`while not (saisie valide):`
Accumuler dans une boucle	`res = 0 while ... : res = res + valeur`
Trouver par doublement	`x = 1 while x < n : x = x * 2`

□□ Toujours modifier une variable dans la boucle — sinon la condition ne change jamais et la boucle tourne à l'infini !

□ **Quand utiliser while et non for ?** Quand le nombre de répétitions dépend d'une réponse de l'utilisateur ou d'un calcul inconnu à l'avance.

□ Bloc 1 — Validation de saisie

Ces exercices représentent le cas d'usage **le plus classique** de while : on ne sait pas combien de fois l'utilisateur va se tromper.

□□ Exercice 1 — Saisie jusqu'à un nombre pair

□ **Objectif :** Redemander un nombre tant que l'utilisateur entre un nombre impair. S'arrêter dès qu'il entre un nombre pair. Utiliser **%** (modulo).

```
terminal
Entrez un nombre :
› 7
Nombre impair, réessaie !
Entrez un nombre :
› 3
Nombre impair, réessaie !
Entrez un nombre :
› 4
Bravo, 4 est pair !
```

□□ Exercice 2 — Saisie dans un intervalle [1, 10]

□ **Objectif :** Demander un entier à l'utilisateur. Recommencer tant que la valeur n'est pas comprise entre 1 et 10. Afficher la valeur validée.

```
terminal
Entrez un entier entre 1 et 10 :
› -3
Valeur invalide, réessaie.
Entrez un entier entre 1 et 10 :
› 15
Valeur invalide, réessaie.
Entrez un entier entre 1 et 10 :
› 7
Valeur acceptée : 7
```

□□ Exercice 3 — Mot de passe

□ **Objectif :** Le bon mot de passe est **python123**. Demander le mot de passe. Recommencer tant qu'il est incorrect. Afficher "Accès autorisé □" dès qu'il est bon.

> □ C'est le cas **classique** de while : on ne peut pas utiliser for car on ne sait pas combien de fois l'utilisateur va se tromper.

```
terminal
Entrez le mot de passe :
› bonjour
```

```
Mot de passe incorrect ■
Entrez le mot de passe :
› 1234
Mot de passe incorrect ■
Entrez le mot de passe :
› python123
Accès autorisé ■
```

□ Bloc 2 — Calculs par progression

Ces exercices utilisent while parce que le **nombre d'étapes dépend du calcul** — impossible de le connaître à l'avance.

□□ Exercice 4 — Première puissance de 2 ≥ n

□ **Objectif :** Lire un entier **n**. Trouver le plus petit exposant **p** tel que $2^p \geq n$. Afficher **p**.

□ Exemple : n = 7 → 2^1=2, 2^2=4, 2^3=8 ≥ 7 → **p = 3**
Exemple : n = 15 → 2^1=2, 2^2=4, 2^3=8, 2^4=16 ≥ 15 → **p = 4**

```
terminal
Entrez n :
› 7
3
```

Autre exemple :

```
terminal
Entrez n :
› 15
4
```

□□ Exercice 5 — Accumuler jusqu'à dépasser 100

□ **Objectif :** Additionner les entiers 1, 2, 3… jusqu'à ce que la somme dépasse 100. Afficher combien d'entiers ont été additionnés et la somme obtenue.

```
terminal
Nombre d'entiers additionnés : 14
```

```
Somme = 105
```

□□ Exercice 6 — Premier multiple commun supérieur à n

□ **Objectif :** Lire deux entiers **a** et **b** et un seuil **n**. Trouver le plus petit entier **m** multiple de a **et** de b, avec m > n.

terminal

```
Entrez a :
› 3
Entrez b :
› 4
Entrez n :
› 10
Premier multiple commun de 3 et 4 supérieur à 10 : 12
```

□ Bloc 3 — Jeu : boucle infinie contrôlée

Ici la boucle tourne **tant que le joueur n'a pas trouvé** — le nombre d'essais est totalement inconnu.

□ Exercice Game — Devinette Magique

□ **Objectif :** Le programme choisit un nombre mystère entre 1 et 20. L'utilisateur a un nombre illimité d'essais. À chaque mauvaise réponse : **Trop petit** ou **Trop grand**. Quand il trouve : **Bravo ! Tu as trouvé en X essais !**

python

```python
from random import randint
mystere = randint(1, 20)
```

terminal

```
Un nombre mystère entre 1 et 20 a été choisi...
Devine :
› 10
Trop grand !
Devine :
› 5
```

```
Trop petit !
Devine :
› 7
Bravo ! Tu as trouvé en 3 essais !
```

□ Récapitulatif — Patterns while

□ Boucle while — cas d'usage

Valider une saisie	`while not (condition valide):`
Mot de passe	`while saisie != mot_de_passe:`
Seuil inconnu	`while x < n : x = x * 2`
Accumuler jusqu'à seuil	`while somme <= 100 : somme = somme + i`
Multiple commun	`while m % a != 0 or m % b != 0 : m = m + 1`
Jeu illimité	`while not trouve :`