

Nom :	Prénom :	Classe :
-------	----------	----------

## SNT — Boucles non bornées

Objectifs :

- Savoir identifier une boucle non bornée.
- Comprendre le rôle et le fonctionnement d'une boucle non bornée.
- Savoir modifier une boucle non bornée.
- Savoir écrire une boucle non bornée.
- Savoir tracer une valeur à travers des itérations.
- Savoir utiliser la commande input() pour récupérer une valeur saisie par l'utilisatrice ou l'utilisateur.

### Introduction

Une boucle non bornée est une **répétition d'instructions** dont on **ne connaît pas le nombre de tours à l'avance** : on répète « tant que *l'affirmation* est vraie ».

Voici quelques exemples de situations dans la vie réelle ressemblant à des boucles non bornées :

- Tant que le feu est rouge, alors je reste à l'arrêt.
- Tant que je n'ai pas trouvé la bonne réponse, alors l'ordinateur me pose une question.
- Tant que j'ai soif, alors je bois de l'eau.

### ► Exercice 1 — Enquêter

Il est possible de représenter les trois exemples ci-dessus dans un tableau.

Complétez le tableau en vous inspirant du premier exemple déjà rempli :

N°	Tant que	Affirmation	Conséquence(s)
1	Tant que	Le feu est rouge	Je reste à l'arrêt
2	Tant que	...	...
3	Tant que	...	...

Avec les boucles **non bornées**, les « conséquences » sont exécutées tant que l'affirmation est vraie.



## Boucle non bornée

Une boucle non bornée **répète** une *ou plusieurs* instructions, en **ne sachant pas**, dès le départ, combien de fois cette répétition aura lieu.

En Python, une boucle non bornée s'écrit de la façon suivante :

```
while affirmation:  
    instruction
```

Ici, l'instruction sera exécutée tant que *affirmation* est vraie.

### Exemple Python

Prenons un exemple, nous souhaitons afficher 10 fois de suite le texte `Bonjour`.

Nous avons vu précédemment comment faire cela avec une **boucle bornée**<sup>1</sup> (ou *boucle for*) :

```
1 for _ in range(10):  
2     print("Bonjour")
```

Et voici à présent comment faire cela avec une **boucle non bornée** (ou *boucle while*) :

```
1 a = 0  
2 while a < 10:  
3     print("Bonjour")  
4     a = a + 1
```

Ces deux exemples (celui de la *boucle for* et celui de la *boucle while*) produisent exactement le **même résultat**, c'est à dire l'affichage de 10 textes Bonjour.



### ► Exercice 2 — Prédire

Lisez le programme suivant :

```
1 age = 1  
2 while age < 18:  
3     print("Vous êtes mineur(e)")  
4     age = age + 1
```

Puis essayez de **deviner** ce que ce programme affichera. Écrivez votre prédiction dans l'espace ci-dessous :

<sup>1</sup> Voir le cours « Boucles bornées ».

### ► Exercice 3 — Vérifier

Exécutez le programme de l'exercice 2, en scannant le QR code ci-dessous :



Alors, votre prédiction était-elle juste ?

### ► Exercice 4 — Modifier

Le programme ci-dessous à gauche affiche un certain nombre de fois le texte "Chocolat".  
Proposez-en une nouvelle version, à droite, en **utilisant une boucle non bornée** :

Un programme Python	Votre programme Python (à compléter)
<pre>print('Chocolat') print('Chocolat') print('Chocolat') print('Chocolat') print('Chocolat') print('Chocolat') print('Chocolat') print('Chocolat') print('Chocolat') print('Chocolat')</pre>	

## Boucle non bornée "non exécutée"

Dans certains cas, il se peut qu'une boucle non bornée ne s'exécute jamais.

Prenons un exemple :

```
1 age = 20
2 while age < 18:
3     print("Vous êtes mineur(e)")
4     age = age + 1
```

Dans le programme ci-dessus, le bloc de code (lignes 3 et 4) n'est jamais exécuté.

Pourquoi ? Car l'**affirmation** de la boucle `age < 18` est **fausse** dès le départ.  
*En effet, la variable age vaut 20 (ligne 1).*

On dit d'une boucle qui n'est pas exécutée qu'elle **n'itère**<sup>2</sup> pas, qu'elle ne boucle pas.



2 Du verbe « itérer » qui signifie « répéter ».

## ► Exercice 5 — Enquêter

Analysez les programmes ci-dessous, et déterminez le **nombre d'itérations** (exemple : 0, 3 ou 18) des boucles qu'ils contiennent :

N°	Programme Python	Nombre d'itérations (à compléter)
1	<pre>a = 123 while a &lt;= 123:     print(a)     a = a + 1</pre>	...
2	<pre>status = False while status == True:     rotate(90)     move_forward(10)</pre>	...
3	<pre>a = 5 b = a * 2 while b &gt; 0:     print(b)     b = b - 1</pre>	...

## Boucle non bornée "infinie"

Dans certains cas, il se peut qu'une boucle non bornée **ne s'arrête jamais**, on parle alors d'une « boucle infinie ».

Une boucle infinie est une boucle non bornée dont **l'affirmation** reste **toujours vraie**, son nombre d'itération est donc ... une infinité de fois !

Prenons un exemple :

```
1 age = 1
2 while age < 18:
3     print("Vous êtes mineur(e)")
```

Dans le programme ci-dessus, le bloc de code (lignes 3) est exécuté une *infinité de fois*.

Pourquoi ? Car **l'affirmation** de la boucle `age < 18` est **vraie**, dès le départ, mais elle reste également vraie, tout au long du programme ! Car *rien*, aucune ligne de code, ne vient modifier la valeur de `age` !

## ► Exercice 6 — Enquêter

Pour chacun des programmes ci-dessous, déterminez :

- Si la boucle itère, ou non.
- Si la boucle est une boucle infinie, ou non.
- Et, si la boucle itère et qu'elle n'est pas infinie, son nombre d'itérations.

La première ligne est fournie en guise d'exemple.

N°	Programme Python	Itère ?	Infinie ?	Itérations
1	<pre>a = 0 while a &lt; 3:     a = a + 1</pre>	OUI	NON	3
2	<pre>a = 0 while a &gt; 0:     a = a + 1</pre>	...	...	...
3	<pre>a = 0 while a &lt; 3:     print(a)</pre>	...	...	...
4	<pre>b = 10 a = b ** 2 while a &gt; 100:     a = a + 1</pre>	...	...	...
5	<pre>age = 1 while age &lt; 18:     print("Encore mineur(e)")     age = age + 1</pre>	...	...	...

## Ce qu'il faut retenir

Vous retrouverez ici *l'essentiel théorique* du cours.

Astuce : transformez ces points en questions/réponses pour vos *flashcards* (application *Anki*).

- Une **boucle non bornée** est une répétition d'instructions dont on ne connaît pas le nombre de tours à l'avance : on répète, on exécute le bloc de code, « tant que l'affirmation est **vraie** ».
- Une **itération** c'est une répétition provoquée par une boucle.
- Une boucle non bornée peut **ne pas itérer une seule fois**, si son **affirmation** est **fausse dès le départ**.
- Une boucle non bornée peut itérer une **infinité de fois**, si son **affirmation** reste **vraie tout au long du programme**. C'est ce qu'on appelle une **boucle infinie**.