TP 3 : Boucles conditionnelles

Description

Objectifs

- ✔ Boucles conditionnelles : `while ... :`
- ✔ Opérateurs de comparaison et opérateurs booléens

Prérequis

- Variables

Événement

https ://wooclap.com

Code : **TFROLC**

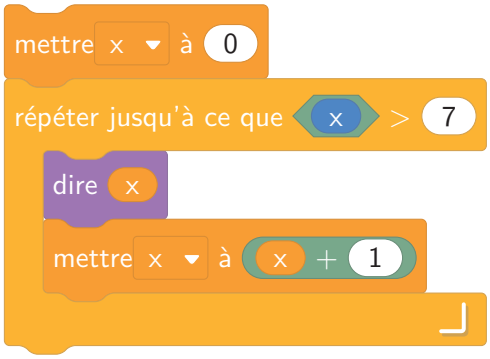
Memo de cours

Instruction `while`

Les boucles conditionnelles (utilisant l'instruction `while`) permettent de répéter un bloc de code tant qu'une condition est vraie.

```
while booléen :
    bloc instructions
```

De Scratch à Python




```
1  x = 0
2  while x <= 7:
3      print(x)
4      x += 1
```

! Entre Scratch et Python, les opérateurs de comparaison sont opposés.

•Opérateurs de comparaison

Les opérateurs de comparaison retournent soit `True` soit `False`.

Test	Sens	Remarques
a == 2	Test d'égalité	Le signe = étant réservé à l'affectation on utilise ici le double égal ==
a != 2	Test de différence	
a > 3.5	Test de stricte supériorité	
a >= 3.5	Test de supériorité large	
a <	Test de stricte infériorité	
a <=	Test d'infériorité large	
"B" in "Bonjour"	Test d'appartenance	Nécessite un itérable

•Opérateurs booléens

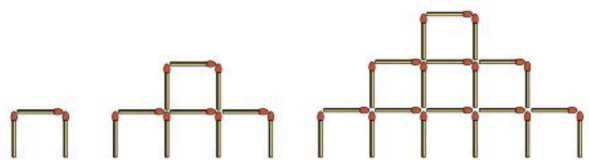
Les opérateurs booléens permettent de lier plusieurs expressions booléennes.

Test	Sens	Remarques
AND	et	retourne `True` si les deux conditions sont vraies et `False` sinon.
OR	ou	retourne `False` si les deux conditions sont fausses et `True` sinon.
NOT	non	transforme `True` en `False` et inversement

Exercices

Exercice 9 Chateau d'allumettes

On s'intéresse à des pyramides construites avec des allumettes comme ci-dessous. En poursuivant ainsi, on obtient des pyramides à autant d'étages que l'on souhaite à condition, bien sûr, d'avoir assez d'allumettes.



Écrire un programme qui permet connaître le nombre maximal d'étages que l'on peut construire avec 1000 allumettes.

Source : Barbazo 1ère

Correction

```python
n = 1
a = 3
s = a
while s < 1000 :
    a += 4
    s += a
    n += 1
# On renvoie l'étape précédente la sortie de la boucle, car on ne pourra pas
↪ construire l'étape n entièrement.
print(n-1)
```

Exercice 10 Algorithme de Babylone

L'algorithme de Babylone, est un algorithme d'approximation de racine carrée.
Pour déterminer une valeur approchée de la racine R du nombre A. On considère les suites (a_n) et (b_n) définie par :

$$\begin{cases} a_0 = A \\ a_{n+1} = \dfrac{a_n + b_n}{2} \end{cases} et\ b_n = \frac{A}{a_n}$$

La suite (a_n) (resp. (b_n)) converge par valeur supérieure (resp. inférieure) vers R
Écrire un programme qui permet d'obtenir une approximation au millième prés de $\sqrt{3}$

Correction

```python
a = 3
b = 1
while a - b > 0.001:
    a = (a+b)/2
    b = 3 / a
print(a)
```

Exercice 11 ★

Dans cet exercice, on considère une feuille de papier A4 que l'on plie en deux.

Déterminer le nombre de pliages nécessaires pour que l'épaisseur d'une feuille de papier dépasse la distance Terre-Lune
Données :

- épaisseur d'une feuille de papier $\approx 0,1$ mm
- Distance Terre-Lune $\approx 384\ 400$ km

Correction

```python
pliage = 0
epaisseur = 1 # en dixième de millimètre
distance_terre_lune = 384400*1000*1000*10 # en dixième de millimètre

while epaisseur < distance_terre_lune:
    epaisseur *= 2
    pliage += 1
print(pliage)
```

Exercice 12 Malthus

i Thomas-Robert Malthus, économiste anglais du début du XIX^{e} siècle, a vécu à une époque où l'industrialisation, couplée aux premières heures d'un marché dominant, ont permises l'expansion économique de la Grande Bretagne.
Durant celle-ci de grands chamboulements sociaux et des nouveaux moyens de productions créèrent autant de richesses que de pauvretés.
Au cours de ces recherches, il a travaillé sur l'évolution de la population en Angleterre.

En 1800, la population anglaise était de 8,3 millions d'habitants.
Bien que très pauvre en majorité, toute la population arrivait tant bien que mal à se nourrir.
On admettra par la suite qu'en 1800 la quantité de nourriture issue de la production agricole permettait de nourrir exactement les 8,3 millions d'habitants.

Thomas Malthus prévoit que cette situation ne pourra pas durer au cours du temps. Il émet les hypothèses suivantes :

— la population en Angleterre augmente chaque année de 2 % ;

— la production agricole anglaise aidée par des avancées techniques, permet de nourrir 400 000 habitants de plus par an.

Écrire un programme Python qui prédit ,selon ce modèle, en quelle année faut-il craindre une famine en Angleterre ?

Correction

```
annee = 1800
population = 8300000
ressources = 8300000
while population <= ressources:
    annee += 1
    population *= 1.02
    ressources += 400000
print(annee)
```