



STRUCTURES CONDITIONNELLES

[Frédéric PEURIERE]

*Connaître les opérateurs de comparaison, les variables de type **bool**, connaître et savoir utiliser les opérateurs d'identité et d'appartenance.*

*Mettre en œuvre des structures conditionnelles avec **if**, **elif** et **else**. Connaître les règles d'entrée et de sortie d'un bloc d'instruction avec Python*

1. LES BOOLEENS:

✓ ACTIVITÉ:

Dans la console: Observez les résultats obtenus puis complétez le tableau « aide mémoire ».

Opérateurs de comparaison: *True* ou *False*?

```
>>> 3>2      >>> 2>3      >>> 3>=2      >>> 2<=2      >>> 3=3      >>> 3==3
>>> 3!=3      >>> 3==3.0      >>> 3!=2      >>> 100/2==50      >>> type(3>2)
```

Opérateurs logiques (*and*, *or* et *not*):

```
>>> a,b,c=1,2,3
>>> a<b and c>b      >>> a<b and a<c      >>> a<b and c<b      >>> a<b or c<b      >>> a>b or c<b
>>> not a>c      >>> not (a<b and c<b)      >>> not a<b and c<b      (not est prioritaire)
>>> jour="mardi"      >>> jour="mardi"      >>> jour=="mardi" or jour=="tuesday"
>>> jour=="mardi" and jour=="tuesday"
```

Les variables de type *bool*:

```
>>> x=3>2      >>> type(x)
```

Opérateurs d'identité (*is*, *is not*):

```
>>> a,b=5,10
>>> a is b      >>> a is not b      >>> 2 is 2.0      >>> 2 == 2.0      >>> a is b/2      >>> a == b/2
```

- L'identité n'est pas une égalité.

Opérateurs d'appartenance (*in*, *not in*):

```
>>> devise="NSI for ever!"      >>> "f" in devise      >>> "n" in devise
>>> "ever" in devise      >>> "ISN" in devise      "er!" in devise      "x" not in devise
>>> " " in devise
```

✓ AIDE MEMOIRE

Opérateurs de comparaison:	
Pour :	Je tape dans la console :
tester si $a > b$ est vrai ou faux	>>>
tester si $a \geq b$ est vrai ou faux	>>>
tester si $a = b$ est vrai ou faux	>>>
tester si $a \neq b$ est vrai ou faux	>>>
tester si $a \leq b$ est vrai ou faux	>>>
Opérateurs logiques:	
tester si $(a > b)$ et $(b = c)$ sont toutes les deux vraies	>>>
tester si au moins une des deux comparaisons est vraie	>>>
inverser le résultat précédent	>>>
Opérateurs d'identité:	
montrer la différence entre l'opérateur de comparaison <code>==</code> et celui d'appartenance <code>is</code>	>>> 5 == 5.0 True >>> False
montrer que les deux variables a et b contiennent deux valeurs différentes	>>> a, b = "Bom", "bom" >>> True
Opérateurs d'appartenance:	
tester la présence d'un espace dans le contenu de la variable <i>bonjour</i>	>>> bonjour = "Bom dia!" >>> True
tester l'absence de point d'exclamation dans le contenu de la variable <i>bonjour</i>	>>> bonjour = "Bom dia" >>> True

2. MISE EN ŒUVRE DES CONDITIONS

✓ ACTIVITÉ:

Avec les conditions testées par **if**, **elif** et **else** nous entrons pour la première fois dans un bloc d'instruction. Avec Python, cette entrée est marquée par le caractère ":" en bout de ligne. Le bloc d'instruction est ensuite *indenté* (c'est à dire décalé) de quatre caractères, ce qui est caractéristique de ce langage. Les autres (Java, C++, JavaScript...) utilisent plutôt des blocs d'instructions entre accolades {}.

Dans THONNY, l'indentation se fait automatiquement lorsqu'un valide par **ENTREE** après le caractère ":" Pour sortir de ce bloc, on tape sur la touche BACKSPACE: **←**. Pour tester les conditions suivantes, passons en **mode programmation**.

Instruction avec *if* (Si) simple:

```
a=1
if a>0:
    print("a est positif")
```

Si la condition est vraie, le programme exécute l'instruction

```
a=1
if a<0:
    print("a est negatif")
```

Si la condition est fausse, le programme ignore l'instruction

```
a="NSI"
if a=="NSI":
    print("NSI!")
```

```
a=12
if a%2==0:
    print(a , « est pair")
```

Utilisation d'opérateurs logiques (*and*, *or* et *not*):

```
a,b=-1,1
if a<0 and b>0:
    print("a<0 et b>0")
```

La condition est vraie dans les deux cas, le programme exécute l'instruction

```
a,b=-1,1
if a<0 and b!=1:
    print("C'est faux!")
```

L'une des deux conditions est fausse, le programme n'exécute pas l'instruction

```
a,b=-1,1
if a<0 or b<0:
    print("Vrai!")
```

La condition est vraie dans au moins un des deux cas, le programme exécute l'instruction

```
a,b="NSI","for ever!"
if not a=="nsi":
    print(a,b)
```

La condition `a=="nsi"` est fausse donc `not a=="nsi"` est vraie, le programme exécute donc l'instruction

Utilisation d'opérateurs d'appartenance et d'identité, introduction de l'alternative *else* (SINON):

```
a="ça va?"  
if "va" in a:  
    print("Vrai!")
```

La variable a contient la chaîne "va", le programme exécute l'instruction

```
a="ça va?"  
if "va!" in a:  
    print("Vrai!")  
else:  
    print("Faux!")
```

La variable a ne contient pas la chaîne "va!", le programme exécute l'instruction alternative (bloc *else*)

```
a="ça va?"  
if a is "ça va":  
    print("Vrai!")  
else:  
    print("Faux!")
```

Les deux chaînes sont différentes, le programme exécute l'instruction alternative (bloc *else*)

```
a="ça va?"  
if a is not "ça va":  
    print("Vrai!")  
else:  
    print("Faux!")
```

Les deux chaînes sont différentes, la condition est donc vraie! le programme exécute l'instruction du bloc *if*

Utilisation d'opérateurs d'appartenance et d'identité, utilisation de l'alternative *elif* (SINON Si).

```
a=int(input("Entrez un entier a:"))  
b=int(input("Entrez un autre entier b:"))  
if b>a:  
    print("b>a")  
elif a==b:  
    print("a=b")  
else:  
    print("a<b")
```

On peut utiliser autant de blocs *elif* que nécessaire mais on termine par un seul bloc *else*.

- Si b est plus grand que a, on exécute seulement l'instruction du bloc **if**: `print("b>a")`

- Si la condition précédente n'est pas vraie évaluons cette nouvelle condition: (`a=b`), on exécute l'instruction du bloc **elif**: `print("a=b")`

- Si elle est fautive également, on exécute la dernière instruction du bloc **else**: `print("a<b")`

✓ AIDE MEMOIRE

Pour:	Je tape en Python:
<p>exécuter des instructions si la condition est vraie, ce qui s'écrit en pseudo code:</p> <pre><i>si condition</i> <i>instructions</i> <i>fin du si</i></pre>	<pre>if condition: instructions</pre>
<p>exécuter les instructions 1 si la condition est vraie. Si elle est fausse, exécuter les instructions 2:</p> <pre><i>si condition</i> <i>instructions 1</i> <i>sinon faire</i> <i>instructions 2</i> <i>fin du si</i></pre>	<pre>if condition: instructions 1 else: instructions 2</pre>
<p>- exécuter les instructions 1 si la condition 1 est vraie - si la condition 1 est fausse mais la condition 2 est vraie, exécuter les instructions 2 - exécuter les instructions 3 dans tous les autres cas:</p> <pre><i>si condition 1</i> <i>instructions 1</i> <i>sinon si condition 2</i> <i>instructions 2</i> <i>sinon</i> <i>instructions 3</i> <i>fin du si</i></pre>	<pre>if condition 1: instructions 1 elif condition 2: instructions 2 else: instructions 3</pre>