T1.2 Valeurs booléennes

Histoire de l'informatique

George Boole (1815-1864) est un mathématicien et logicien britannique connu pour avoir créé la logique moderne, appelée *algèbre de Boole*.

Cette algèbre binaire n'accepte que deux valeurs, 0 et 1, et a donc d'importantes et nombreuses applications en informatique...

1. 1.2.1 Un peu de logique

En informatique, comme en mathématiques, on s'intéresse à la valeur de vérité de phrases ou d'expressions qui peuvent être soit vraies, soit fausses. Mais rien d'autre, c'est le principe du tiers-exclu.

Par exemple, que diriez-vous de ces phrases?

- A: Vous êtes en classe de première.
- B: Baudelaire a écrit «Les fleurs du mal».
- · C: La Terre est plate.
- D: $3 \times 4 = 12$.
- E: La lettre e est dans le mot abracadabra.
- F: Georges Perec a écrit un roman de près de 300 pages sans aucune lettre e.
- G: $2^{10} < 10^3$
- H: La couleur orange est la plus belle des couleurs.

2. 1.2.2 Algèbre de Boole

Valeurs et opérations fondamentales

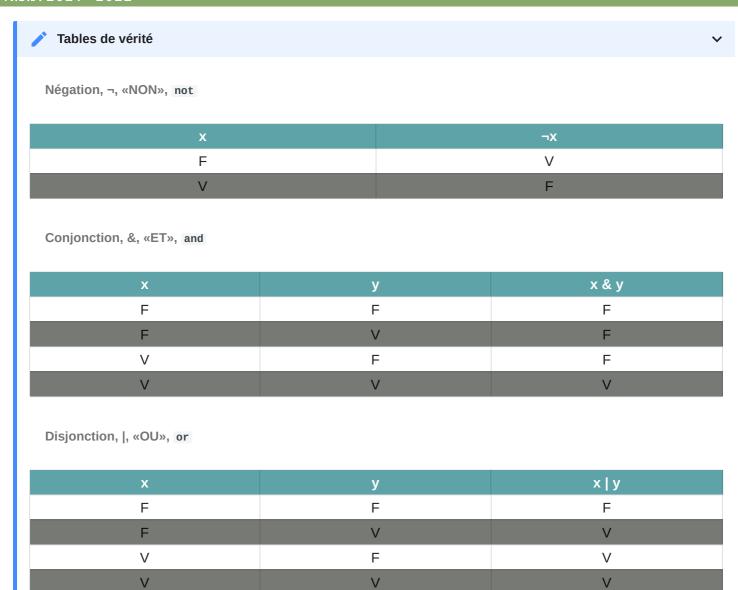
L'algèbre de Boole consiste à étudier des opérations sur un ensemble uniquement constitué de deux éléments qu'on appelle booléens.

Selon le contexte (logique, calcul, électronique), ces deux éléments sont notés:

- Faux (F) / Vrai (V)
- 0/1
- False / True (en Python, comme dans de nombreux langages)

Les opérations fondamentales ne sont plus l'addition et la multiplication mais:

- la **négation**, notée ¬, ou plus simplement «NON» (not en Python);
- la conjonction, notée &, ou plus simplement «ET» (and en Python);
- la disjonction, notée |, ou plus simplement «OU» (or en Python).



3. 1.2.3 Avec Python

True & False

- Il existe deux valeurs booléennes en Python : True et False.
- Une variable prenant l'une de ces deux valeurs est de type bool.

% Script Python

```
>>> type(True)
<class 'bool'>
>>> x = False
>>> x
False
>>> type(x)
<class 'bool'>
```

i Opérateurs de comparaison

Opérateur	Signification
==	est égal à
Œ	est différent de
<	inférieur à
>	supérieur à
<=	inférieur ou égal à
>=	supérieur ou égal à
in	appartient à
not in	n'appartient pas à

Exemples

% Script Python

```
>>> a = 2
>>> a == 3
False
>>> a == 2
True
>>> a != 1
True
>>> a > 2
False
>>> a <= 5
True
>>> a % 2 == 0
True
>>> x = (0 == 1)
>>> X
False
>>> y = (3 + 2 == 5)
>>> y
True
>>> 'e' in 'abracadabra'
False
>>> 'b' in 'abracadabra'
True
>>> 'A' not in 'abracadabra'
```

4. 1.2.4 Exercices

True

False

False

True

>>> not True

>>> True and False

>>> True and True

>>> False or True

Exercice 1

Énoncé

Prédire si les variables suivantes contiennent le booléen True ou le booléen False. Contrôlez ensuite en exécutant le code et en inspectant le contenu des variables.

& Script Python

```
a = (2 > 1)
b = (3 == 1+2)
c = (1 < 0)
d = (2 != 5/2)
e = (2 != 5//2)
f = ('a' == 'A')
g = not a
h = b and c
i = b or c
j = not c and (d or e)</pre>
```

Correction

Exercice 2 : le «ou exclusif»

Énoncé

Une autre fonction logique importante est le ou exclusif, ou «disjonction exclusive».

C'est le «ou» dans le sens de «Fromage *ou* dessert» dans un menu au restaurant. Soit l'un, soit l'autre, mais pas les deux.

Il se note en général xor ou ^ en logique (^ en Python).

Si x et y sont deux booléens, alors $x \wedge y = (x \& \neg y) \mid (\neg x \& y)$.

Construire la table de vérité du xor.

Correction

Exercice 3

Énoncé

Construire la table de vérité de l'expression (x | y) & z où x, y et z sont trois booléens.

Correction

Exercice 4

Énoncé

À l'aide de tables de vérité, démontrer les lois de Morgan:

•
$$\neg(x \mid y) = \neg x \& \neg y$$

•
$$\neg(x \& y) = \neg x \mid \neg y$$

Correction