



Algèbre de Boole

Un peu d'histoire

George Boole (1847)

En 1847, le britannique **George Boole** inventa un formalisme permettant d'écrire des raisonnements logiques : l'**algèbre de Boole**.

Cette algèbre ne contient que **deux valeurs** et **3 opérations** de base.

Claude Shannon (1938)

En 1938, l'américain **Claude Shannon** prouva dans sa thèse que des circuits électriques peuvent résoudre tous les problèmes que l'algèbre de Boole peut elle-même résoudre.

La route était ouverte à l'apparition des premiers ordinateurs électroniques.

Les booléens et les opérateurs fondamentaux

Les booléens

Les booléens ne possèdent que **deux valeurs**, codées sur un seul bit :

- **FAUX** : noté 0, False, false ...
- **VRAI** : noté 1, True, true ...

Introduction à l'algèbre de Boole

L'algèbre de Boole repose sur **3 opérateurs fondamentaux** :

- l'opérateur **NON**
- l'opérateur **ET**
- l'opérateur **OU**

Puisqu'il n'y a que 2 valeurs possibles, on peut dresser la liste de tous les cas dans un tableau appelé **table de vérité**.

NON (NOT)

 NON – notations

- NON a
- $\neg a$ (logique mathématique)
- `not a` (Python)
- `!a` (C, Java...)
- \bar{a} (électronique)

 Exercice 1 – 1 – Table de vérité de NON

Compléter la table de vérité de $\neg a$:

a	$\neg a$
0	
1	



Réponse

ET (AND)

 ET – notations

- a ET b
- $a \wedge b$ (logique mathématique)
- `a and b` (Python)

- $a \cdot b$ (mathématiques)
- $a \& b$ (bit à bit)

Exercice 2 – 2 – Table de vérité de ET

Compléter la table de vérité de $a \wedge b$:

a	b	$a \wedge b$
0	0	
0	1	
1	0	
1	1	



Réponse



Exercice 3 – 3 – Python et AND

Évaluer chaque expression Python :

 Python

```
>>> (1 + 2 == 3) and (7 + 1 ==  
↪ 8)  
>>> (1 + 2 == 3) and (1 + 1 ==  
↪ 3)
```

Puis expliquer le résultat ci-dessous :

 Python

```
>>> (7 / 0 == 1) and (4 / 2 ==  
↪ 2)  
>>> (1 + 1 == 3) and (7 / 0 ==  
↪ 2)
```



Réponse

.....

.....

.....

.....

OU (OR)

 OU – notations

- $a \text{ OU } b$
- $a \vee b$ (logique mathématique)
- **a or b** (Python)
- $a + b$ (mathématiques)
- **a | b** (bit à bit)

 Exercice 4 – 4 – Table de vérité de OU

Compléter la table de vérité de $a \vee b$:

a	b	$a \vee b$
0	0	
0	1	
1	0	
1	1	



Réponse

 Exercice 5 – 5 – Python et OR Python

```
>>> n = 20
>>> (n % 10 == 0) or (n % 7 ==
↪ 0)
>>> (n % 4 == 0) or (n % 5 == 0)
>>> (n % 7 == 0) or (n % 3 == 0)
```

Expliquer le résultat :

 Python

```
>>> n = 20
>>> (n % 5 == 0) or (n % 0 == 0)
```

 Réponse

.....

.....

.....

Fonctions composées

XOR – OU exclusif

XOR

Vrai uniquement si les deux opérandes sont **différents**.

$$x \oplus y = (x \wedge \neg y) \vee (\neg x \wedge y)$$

– $a \text{ XOR } b$

– $a \oplus b$ (logique mathématique)

– $a \wedge b$ (Python, bit à bit)

Exercice 6 – 6 – Table de XOR

Compléter toutes les colonnes :

a	b	$a \wedge \neg b$	$\neg a \wedge b$	$a \oplus b$
0	0			
0	1			
1	0			
1	1			



Réponse

NAND – NON ET

NAND

$$x \uparrow y = \neg(x \wedge y)$$

- $a \text{ NAND } b$
- $\neg(a \wedge b)$
- `not(a and b)` (Python)

Exercice 7 – 7 – Table de NAND


Retrouver la table en complétant les colonnes intermédiaires :

a	b	$a \wedge b$	$\neg(a \wedge b)$
0	0		
0	1		
1	0		
1	1		



Réponse

NOR – NON OU

 NOR

$$x \downarrow y = \neg(x \vee y)$$

- $a \text{ NOR } b$
- $\neg(a \vee b)$
- `not(a or b)` (Python)

 Exercice 8 – 8 – Table de NOR

Retrouver la table en complétant les colonnes intermédiaires :

a	b	$a \vee b$	$\neg(a \vee b)$
0	0		
0	1		
1	0		
1	1		



Réponse

Exercice 9 – 9 – Opérations bit à bit

Effectuer les opérations suivantes en binaire, puis vérifier en Python :

`</>` AND bit à bit `</>` OR bit à bit `</>` XOR bit à bit

$$\begin{array}{r} \hookrightarrow 10110 \\ \& \\ \hookrightarrow 1101010 \\ \hline \end{array}$$

?

$$\begin{array}{r} \hookrightarrow 10110 \\ | \\ \hookrightarrow 1010101 \\ \hline \end{array}$$

?

$$\begin{array}{r} \hookrightarrow 1011011 \\ \wedge \\ \hookrightarrow 1010101 \\ \hline \end{array}$$

?



Réponse

.....

.....

.....

Lois de Morgan

Lois De Morgan

Pour tous booléens a et b :

- **1^{er} théorème** : $\neg(a \vee b) = \neg a \wedge \neg b$
- **2^e théorème** : $\neg(a \wedge b) = \neg a \vee \neg b$

Exercice 10 – 11 – Démonstration par table de

Compléter les deux tables pour démontrer le **1^{er} théorème de De Morgan**.

Table de $\neg(a \vee b)$:

a	b	$a \vee b$	$\neg(a \vee b)$
0	0		
0	1		
1	0		
1	1		

Table de $\neg a \wedge \neg b$:

a	b	$\neg a$	$\neg b$	$\neg a \wedge \neg b$
0	0			
0	1			
1	0			
1	1			



Réponse



**Exercice 11 – 12 – Distributivité**

Soient a , b et c trois booléens. Établir les tables de vérité de :

– $a \wedge (b \vee c)$ (a ET (b OU c))

– $(a \wedge b) \vee (a \wedge c)$ ((a ET b) OU (a ET c))

Ces deux expressions sont-elles équivalentes ?



Réponse

.....

.....

.....

.....

.....

.....