



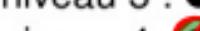
## Information, Calcul et Communication Compléments de cours

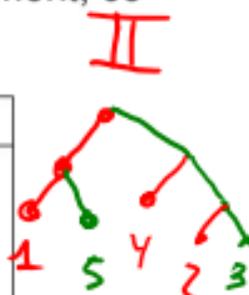
J.-C. Chappelier

## Leçon II.1 et II.2 – Examen final 2018 1.3



Des séquences de cinq niveaux d'alerte météo doivent être transmises codées (code sans-préfixe et sans perte) sous forme de séquences de pastilles (ronds) rouges ou vertes. La table ci-dessous représente trois propositions de codes possibles. Malheureusement, ce sujet est tiré en noir et blanc ; la couleur verte ou rouge s'est donc perdue...

	code I	code II	code III
niveau 1 :			
niveau 2 :			
niveau 3 :			
niveau 4 :			
niveau 5 :			



Quel(s) code(s) (d'origine, avec les couleurs) êtes vous néanmoins sûr(e) de ne pas pouvoir utiliser pour la transmission désirée ?

Justifiez votre réponse.

$$\sum_i 2^{-l_i} \leq 1$$

## Leçon II.4 – Examen final 2018 1.2

À partir d'un alphabet de 33 lettres, on compose un mot  $X$  de 128 lettres de long ; chacune des lettres de l'alphabet étant présente au moins une fois dans le mot  $X$ . Le code de Huffman de ce mot a une longueur moyenne de 5.5 bits.

1. Est-ce possible ? **Justifiez** votre réponse.

oui non  
krp 2

2. Si **oui**, donnez les *meilleures* bornes (haute et basse) que vous pouvez pour

2.1 l'entropie de ce mot :

2.2 la longueur moyenne d'un code de Shannon-Fano de ce mot :

$$4.5 = L_H - 1 \left. \begin{array}{l} H(X_{\min}) \\ H(X) \end{array} \right\} \leq H(X) \leq \log_2 33 \approx \log_2 32 < 5.5 < 6$$
$$L_H \leq L_c(\text{Shannon-Fano}(X)) \leq H(X) + 1 \quad \begin{array}{l} 5 \\ 5+1 \end{array}$$

et si c'est **non**, donnez les *meilleures* bornes (haute et basse) que vous pouvez pour la longueur moyenne d'un code de Huffman de ce mot :

$$\leq L_c(\text{Huffman}(X)) \leq$$

$$0 \leq H(x) \leq \log_2 n \rightarrow \text{alphabet}$$

$\forall$  code  $C \dots$

$$H(x) \leq L_C$$

$\exists$  code  $C'$

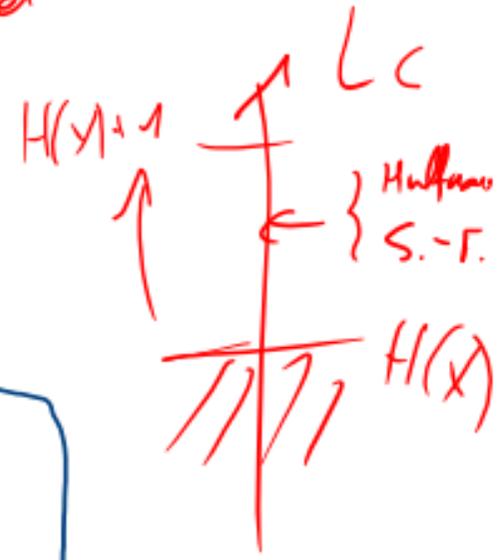
$$L_{C'} \leq \underbrace{H(x) + 1}_{H_{\min}}$$

fluffman

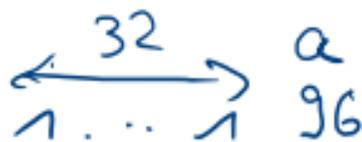
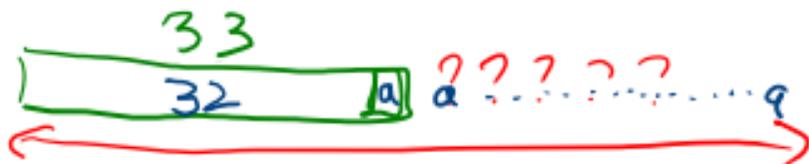
$\forall$  code  $C \dots$

$$L_{\text{fluff}} \leq L_C$$

$$L_{C'} - 1 < H(x)$$



$X_{\min}$  ?

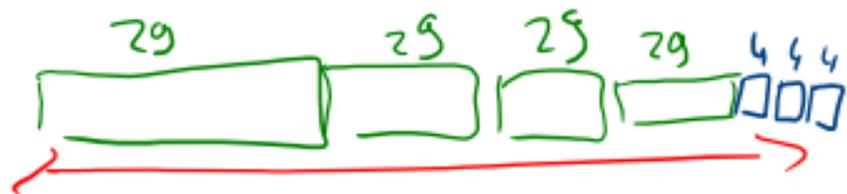


128

$$\underbrace{\log_2 128}_7 - \frac{96}{128} \log_2 96$$

$H(X_{\min})$

~~X~~ max



128

33

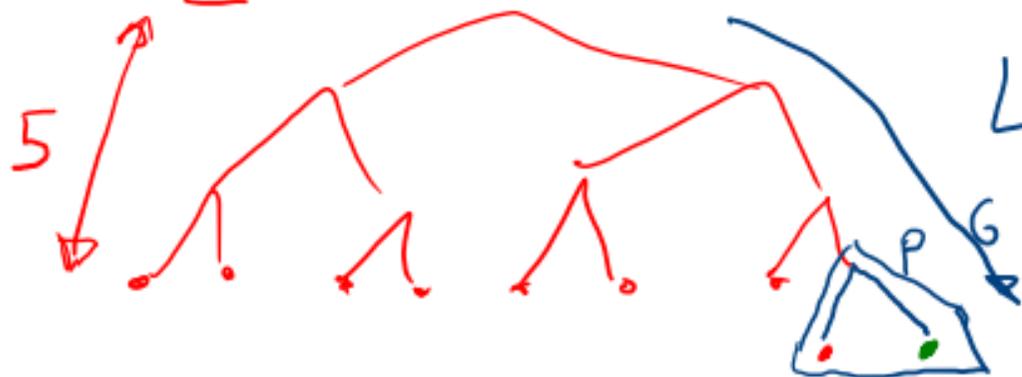
12

$\frac{128}{32} = 4$



$$29 \times 4 + 4 \times 3 = 128$$

Huffman 128 32 letters  $\rightarrow$  binary exact



$$L_H = p \cdot 6 + (1-p) \cdot 5 = 5 + p$$

## Leçon II.4 – Examen final 2018 1.2

À partir d'un alphabet de 33 lettres, on compose un mot  $X$  de 128 lettres de long ; chacune des lettres de l'alphabet étant présente au moins une fois dans le mot  $X$ . Le code de Huffman de ce mot a une longueur moyenne de 5.5 bits.

1. Est-ce possible ? **Justifiez** votre réponse.  
*oui ça semble possible (réponse attendue)...*
2. Si **oui**, donnez les *meilleures* bornes (haute et basse) que vous pouvez pour
  - 2.1 l'entropie de ce mot :

$$\leq H(X) \leq$$

- 2.2 la longueur moyenne d'un code de Shannon-Fano de ce mot :

$$\leq L_c(\text{Shannon-Fano}(X)) \leq$$

et si c'est **non**, donnez les *meilleures* bornes (haute et basse) que vous pouvez pour la longueur moyenne d'un code de Huffman de ce mot :

$$\leq L_c(\text{Huffman}(X)) \leq$$

## Leçon II.4 – Examen final 2018 1.2

À partir d'un alphabet de 33 lettres, on compose un mot  $X$  de 128 lettres de long ; chacune des lettres de l'alphabet étant présente au moins une fois dans le mot  $X$ . Le code de Huffman de ce mot a une longueur moyenne de 5.5 bits.

1. Est-ce possible ? **Justifiez** votre réponse.  
*oui ça semble possible (réponse attendue)... ...mais en fait, c'est impossible*
2. Si **oui**, donnez les *meilleures* bornes (haute et basse) que vous pouvez pour
  - 2.1 l'entropie de ce mot :

$$\leq H(X) \leq$$

- 2.2 la longueur moyenne d'un code de Shannon-Fano de ce mot :

$$\leq L_c(\text{Shannon-Fano}(X)) \leq$$

et si c'est **non**, donnez les *meilleures* bornes (haute et basse) que vous pouvez pour la longueur moyenne d'un code de Huffman de ce mot :

$$\leq L_c(\text{Huffman}(X)) \leq$$

## Leçon II.4 – Examen final 2018 1.2

À partir d'un alphabet de 33 lettres, on compose un mot  $X$  de 128 lettres de long ; chacune des lettres de l'alphabet étant présente au moins une fois dans le mot  $X$ . Le code de Huffman de ce mot a une longueur moyenne de 5.5 bits.

1. Est-ce possible ? **Justifiez** votre réponse.  
*oui ça semble possible (réponse attendue)... ...mais en fait, c'est impossible*
2. Si **oui**, donnez les *meilleures* bornes (haute et basse) que vous pouvez pour
  - 2.1 l'entropie de ce mot :

$$\leq H(X) \leq$$

- 2.2 la longueur moyenne d'un code de Shannon-Fano de ce mot :

$$\leq L_c(\text{Shannon-Fano}(X)) \leq$$

et si c'est **non**, donnez les *meilleures* bornes (haute et basse) que vous pouvez pour la longueur moyenne d'un code de Huffman de ce mot :

$$H(X) \leq L_c(\text{Huffman}(X)) \leq 5.047 \text{ (voir plus loin)}$$

## Leçon II.4 – Examen final 2018 1.2

À partir d'un alphabet de 33 lettres, on compose un mot  $X$  de 128 lettres de long ; chacune des lettres de l'alphabet étant présente au moins une fois dans le mot  $X$ . Le code de Huffman de ce mot a une longueur moyenne de 5.5 bits.

1. Est-ce possible ? **Justifiez** votre réponse.  
*oui ça semble possible (réponse attendue)... ...mais en fait, c'est impossible*
2. Si **oui**, donnez les *meilleures* bornes (haute et basse) que vous pouvez pour
  - 2.1 l'entropie de ce mot :

$$L_c(\text{Huffman}(X)) - 1 \leq H(X) \leq$$

- 2.2 la longueur moyenne d'un code de Shannon-Fano de ce mot :

$$\leq L_c(\text{Shannon-Fano}(X)) \leq$$

et si c'est **non**, donnez les *meilleures* bornes (haute et basse) que vous pouvez pour la longueur moyenne d'un code de Huffman de ce mot :

$$H(X) \leq L_c(\text{Huffman}(X)) \leq 5.047 \text{ (voir plus loin)}$$

## Leçon II.4 – Examen final 2018 1.2

À partir d'un alphabet de 33 lettres, on compose un mot  $X$  de 128 lettres de long ; chacune des lettres de l'alphabet étant présente au moins une fois dans le mot  $X$ . Le code de Huffman de ce mot a une longueur moyenne de 5.5 bits.

1. Est-ce possible ? **Justifiez** votre réponse.  
*oui ça semble possible (réponse attendue)... ...mais en fait, c'est impossible*
2. Si **oui**, donnez les *meilleures* bornes (haute et basse) que vous pouvez pour
  - 2.1 l'entropie de ce mot :

$$L_c(\text{Huffman}(X)) - 1 \leq H(X) \leq \log_2(33) \simeq 5.044$$

- 2.2 la longueur moyenne d'un code de Shannon-Fano de ce mot :

$$\leq L_c(\text{Shannon-Fano}(X)) \leq$$

et si c'est **non**, donnez les *meilleures* bornes (haute et basse) que vous pouvez pour la longueur moyenne d'un code de Huffman de ce mot :

$$H(X) \leq L_c(\text{Huffman}(X)) \leq 5.047 \text{ (voir plus loin)}$$

## Leçon II.4 – Examen final 2018 1.2

À partir d'un alphabet de 33 lettres, on compose un mot  $X$  de 128 lettres de long ; chacune des lettres de l'alphabet étant présente au moins une fois dans le mot  $X$ . Le code de Huffman de ce mot a une longueur moyenne de 5.5 bits.

1. Est-ce possible ? **Justifiez** votre réponse.  
*oui ça semble possible (réponse attendue)... ...mais en fait, c'est impossible*
2. Si **oui**, donnez les *meilleures* bornes (haute et basse) que vous pouvez pour
  - 2.1 l'entropie de ce mot :

$$L_c(\text{Huffman}(X)) - 1 \leq H(X) \leq \dots \text{ ou : } \frac{166}{32} - \frac{3}{32} \log_2(3) \simeq 5.039$$

- 2.2 la longueur moyenne d'un code de Shannon-Fano de ce mot :

$$\leq L_c(\text{Shannon-Fano}(X)) \leq$$

et si c'est **non**, donnez les *meilleures* bornes (haute et basse) que vous pouvez pour la longueur moyenne d'un code de Huffman de ce mot :

$$H(X) \leq L_c(\text{Huffman}(X)) \leq 5.047 \text{ (voir plus loin)}$$

## Leçon II.4 – Examen final 2018 1.2

À partir d'un alphabet de 33 lettres, on compose un mot  $X$  de 128 lettres de long ; chacune des lettres de l'alphabet étant présente au moins une fois dans le mot  $X$ . Le code de Huffman de ce mot a une longueur moyenne de 5.5 bits.

1. Est-ce possible ? **Justifiez** votre réponse.  
*oui ça semble possible (réponse attendue)... ...mais en fait, c'est impossible*
2. Si **oui**, donnez les *meilleures* bornes (haute et basse) que vous pouvez pour
  - 2.1 l'entropie de ce mot :

$$L_c(\text{Huffman}(X)) - 1 \leq H(X) \leq \dots \text{ ou : } \frac{166}{32} - \frac{3}{32} \log_2(3) \simeq 5.039$$

- 2.2 la longueur moyenne d'un code de Shannon-Fano de ce mot :

$$5.5 \leq L_c(\text{Shannon-Fano}(X)) \leq$$

et si c'est **non**, donnez les *meilleures* bornes (haute et basse) que vous pouvez pour la longueur moyenne d'un code de Huffman de ce mot :

$$H(X) \leq L_c(\text{Huffman}(X)) \leq 5.047 \text{ (voir plus loin)}$$

## Leçon II.4 – Examen final 2018 1.2

À partir d'un alphabet de 33 lettres, on compose un mot  $X$  de 128 lettres de long ; chacune des lettres de l'alphabet étant présente au moins une fois dans le mot  $X$ . Le code de Huffman de ce mot a une longueur moyenne de 5.5 bits.

1. Est-ce possible ? **Justifiez** votre réponse.  
*oui ça semble possible (réponse attendue)... ...mais en fait, c'est impossible*
2. Si **oui**, donnez les *meilleures* bornes (haute et basse) que vous pouvez pour
  - 2.1 l'entropie de ce mot :

$$L_c(\text{Huffman}(X)) - 1 \leq H(X) \leq \dots \text{ ou : } \frac{166}{32} - \frac{3}{32} \log_2(3) \simeq 5.039$$

- 2.2 la longueur moyenne d'un code de Shannon-Fano de ce mot :

$$5.5 \leq L_c(\text{Shannon-Fano}(X)) \leq H(X) + 1$$

et si c'est **non**, donnez les *meilleures* bornes (haute et basse) que vous pouvez pour la longueur moyenne d'un code de Huffman de ce mot :

$$H(X) \leq L_c(\text{Huffman}(X)) \leq 5.047 \text{ (voir plus loin)}$$

## Leçon II.4 – Réponse

1. Est-ce possible ? **Justifiez** votre réponse.

Réponse attendue : oui ça *semble* possible :

$$H(X) \leq \log_2(33) < L_c(\text{Huffman}(X))$$

et :  $L_c(\text{Huffman}(X)) < \log_2(33) + 1$  (utiliser  $\log_2(32) = 5$ ,  $\log_2(33) = 5 + \log_2(33/32)$ )

Ceci dit, le choix de 5.5 pour  $L_c(\text{Huffman}(X))$  est un peu extrême et, en fait, trop grand. On pourrait par exemple le majorer par un code (pas forcément optimal) ; p.ex. 31 à 5 bits, et deux à 6 bits

lequel donne une longueur moyenne de  $5 + p$  (avec  $p$  la somme des probabilités des deux à 6 bits ; donc  $p$  compris entre  $\frac{2}{128}$  et  $\frac{6}{128}$  [sinon on aurait un meilleur code en en mettant d'autres à 6 bits :  $128 - 33n_{max} \geq 0 : 128/33 \geq n_{max}$ ]),

donc une longueur moyenne de ce code entre 5.016 bits et 5.047 bits

qui donne un majorant inférieur à 5.5 : donc c'est, en fait, impossible d'avoir 5.5 (puisque le code de Huffman est optimal)

Mais je n'attends pas un tel niveau de raisonnement en examen en temps limité.

## Leçon II.4 – Réponse

Si on veut faire l'étude complète de ce cas (mais veut-on la faire en examen ?), la situation varie entre

- ▶ 29 lettres apparaissent 4 fois et les 4 autres lettres apparaissent 3 fois ;

ce qui fait une entropie de  $\frac{29}{32} \times 5 + \frac{3}{32} \log_2\left(\frac{128}{3}\right) \simeq 5.039$  bit.

et une longueur moyenne du code de Huffman de  $\frac{646}{128} \simeq 5.047$  bits  
(31 fois 5 et 2 fois 6, donc exemple précédent avec  $p = \frac{6}{128}$ )

(pour info  $\log_2(33) \simeq 5.044$ )

- ▶ 32 lettres apparaissent 1 fois et la dernière lettre apparait 96 fois,

ce qui fait une entropie de  $\frac{1}{4} \times 7 + \frac{3}{4} \log_2\left(\frac{4}{3}\right) \simeq 2.061$  bit

et une longueur moyenne du code de Huffman de  $\frac{288}{128} \simeq 2.25$  bits

## Leçon III.1 (architecture des ordinateurs) – Points clés

- ▶ architecture de von Neumann :  
processeur (CPU), mémoire, périphériques
- ▶ composants d'un CPU :  
registres, ALU, (Décodeur, ) (pointeur de pile, ) (contrôleur )  
réalisés à l'aide de **transistors**
- ▶ mémoire : 2 inverseurs « tête-bêche » (= 4 transistors)
- ▶ compilation :
  - ▶ assembleur : registres, instruction (dont comparaisons et sauts)
  - ▶ langage machine : encodage binaire de l'assembleur (y compris opérandes)
- ▶ performances / énergie : jouer sur :
  - ▶ le délai ;
  - ▶ et le débit (parallélisme).

## Leçon III.1 (Architecture des ordinateurs) – Étude de cas

Considérez le code assembleur suivant :

```
1: charge r2, 0
2: charge r3, 0
3: charge r4, r3
4: somme r3, r3, 1
5: somme r4, r4, 1
6: cont_ppe r1, r4, 9
7: somme r2, r2, r4
8: continue 5
9: somme r2, r2, r3
10: cont_pp r3, r0, 3
```

où l'instruction « cont\_ppe a, b, N » effectue le test «  $a \leq b$  »  
et l'instruction « cont\_pp a, b, N » effectue le test «  $a < b$  ».  
Lequel de ces algorithmes correspond au code ci-dessus  
(avec  $n$  chargé dans  $r0$  et  $m$  dans  $r1$ ) :

```
r0 ← m
r1 ← m
r2 ← 0
r3 ← 0
do
  r4 ← r3
  r3 ← r3 + 1
  do
    r4 ← r4 + 1
  while (true);
  r2 ← r2 + r3
while (r3 < r0);
```

$r_1 > r_4$   
NEG ↑  
if ( $r_1 \leq r_4$ )  
break;  
 $r_2 \leftarrow r_2 + r_4$   
NEG  
→  $r_3 > r_0$

$$\underbrace{\sum_{\substack{m \\ R_3 = 1}} \left( \sum_{\substack{m-1 \\ R_4 = R_3}} R_4 \right) + R_3}$$

A.

<b>algoA</b>
entrée : deux entiers $1 \leq n \leq m$
sortie : S
$S \leftarrow 0$ $i \leftarrow 0$ <b>Tant que</b> $i < n$ $j \leftarrow i$ <b>Tant que</b> $j < m$ <b>Si</b> $j < m$ $S \leftarrow S + j$ <b>Sinon</b> $S \leftarrow S + i$ Sortir : S

B.

<b>algoB</b>
entrée : deux entiers $1 \leq n \leq m$
sortie : S
$S \leftarrow 0$ <b>Pour</b> $i$ de $n$ à 1 en descendant (de 1 en 1) <b>Pour</b> $j$ de $m-1$ à $i$ en descendant (de 1 en 1) $S \leftarrow S + j$ $S \leftarrow S + i$ Sortir : S

C.

<b>algoC</b>
entrée : <i>deux entiers</i> $1 \leq n \leq m$ sortie : $S$
$S \leftarrow 0$ <b>Pour</b> $i$ de 1 à $n-1$   <b>Pour</b> $j$ de 1 à $m$     $S \leftarrow S+j$     $S \leftarrow S+i$ <b>Sortir</b> : $S$

D.

<b>algoD</b>
entrée : <i>deux entiers</i> $1 \leq n \leq m$ sortie : $S$
<b>Si</b> $n = 1$   <b>Sortir</b> : $1 + \frac{(m-1) \cdot m}{2}$ $s \leftarrow n$ <b>Pour</b> $i$ de $m-1$ à $n$ en descendant (de 1 en 1)   $s \leftarrow s+i$ <b>Sortir</b> : $s + \text{algoD}(n-1, m)$

C.

<b>algoC</b>
entrée : <i>deux entiers</i> $1 \leq n \leq m$ sortie : $S$
$S \leftarrow 0$ <b>Pour</b> $i$ de 1 à $n-1$   <b>Pour</b> $j$ de 1 à $m$     $S \leftarrow S+j$     $S \leftarrow S+i$ <b>Sortir</b> : $S$

\*D.

<b>algoD</b>
entrée : <i>deux entiers</i> $1 \leq n \leq m$ sortie : $S$
<b>Si</b> $n = 1$   <b>Sortir</b> : $1 + \frac{(m-1) \cdot m}{2}$ $s \leftarrow n$ <b>Pour</b> $i$ de $m-1$ à $n$ en descendant (de 1 en 1)   $s \leftarrow s+i$ <b>Sortir</b> : $s + \text{algoD}(n-1, m)$

## Leçon III.1 (Architecture des ordinateurs) – Réponse

Dans l'algorithme A, il manque les incréments de  $i$  et  $j$  dans leur boucle respective (boucles infinies).

Dans l'algorithme B, les deux lignes «  $S \leftarrow S + i$  » et «  $S \leftarrow S + j$  » ont été inversées (aucun sens !).

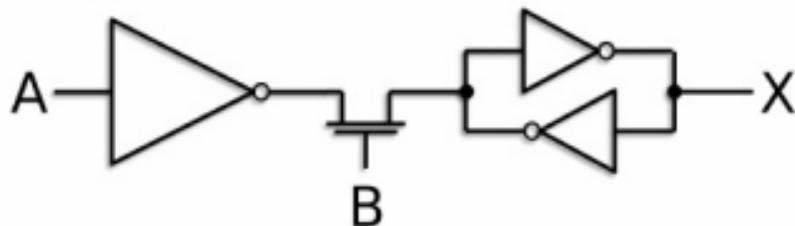
Les bornes de l'algorithme C ne sont pas correctes : la boucle en  $i$  devrait aller jusque  $n$  et celle en  $j$  de  $i$  à  $m - 1$  (à voir en testant le cas simple  $n = 1$ ).

L'algorithme D est une écriture récursive de ce que calcule le programme :

$$S(n, m) = \sum_{i=1}^n \left( \left( \sum_{j=i}^{m-1} j \right) + i \right) = \sum_{i=1}^{n-1} \left( \left( \sum_{j=i}^{m-1} j \right) + i \right) + \sum_{j=n}^{m-1} j + n = n + \sum_{j=n}^{m-1} j + S(n-1, m)$$

## Leçon III.1 (Architecture des ordinateurs) – Étude de cas

On considère le système logique suivant :



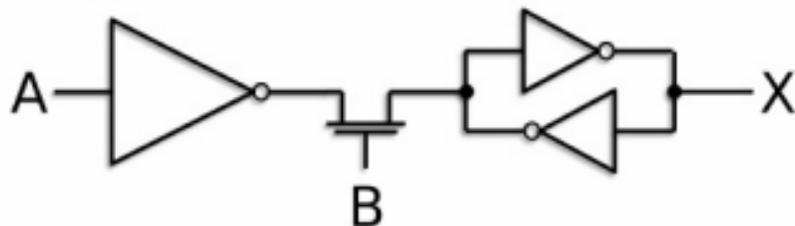
auquel on soumet les entrées A et B suivantes à quatre instants consécutifs :

t	A	B	X
1	0	1	$x_1$
2	1	0	$x_2$
3	0	1	$x_3$
4	1	1	$x_4$

Quelles sont les quatre sorties ( $x_1, x_2, x_3, x_4$ ) correspondantes ?

## Leçon III.1 (Architecture des ordinateurs) – Étude de cas

On considère le système logique suivant :



auquel on soumet les entrées A et B suivantes à quatre instants consécutifs :

t	A	B	X
1	0	1	$x_1$
2	1	0	$x_2$
3	0	1	$x_3$
4	1	1	$x_4$

Quelles sont les quatre sorties ( $x_1, x_2, x_3, x_4$ ) correspondantes ?

(0, 0, 0, 1)

## Leçon III.1 (Architecture des ordinateurs) – Étude de cas

Quelle est la table de vérité du programme ci-contre (sachant que r1 et r2 contiennent soit 0 soit 1) ?

**A]**

r1	r2	r3
0	0	0
1	0	0
0	1	0
1	1	10

**B]**

r1	r2	r3
0	0	0
1	0	1
0	1	1
1	1	1

**C]**

r1	r2	r3
0	0	0
1	0	1
0	1	1
1	1	0

**D]** Aucune des trois.

```
1: cont_egal r1, 0, 5
2: cont_egal r2, 0, 5
3: charge r3, 0
4: continue 6
5: somme r3 r1 r2
6: // fin (stop)
```

## Leçon III.1 (Architecture des ordinateurs) – Étude de cas

Quelle est la table de vérité du programme ci-contre (sachant que r1 et r2 contiennent soit 0 soit 1) ?

A]

r1	r2	r3
0	0	0
1	0	0
0	1	0
1	1	10

B]

r1	r2	r3
0	0	0
1	0	1
0	1	1
1	1	1

C] \*

r1	r2	r3
0	0	0
1	0	1
0	1	1
1	1	0

D] Aucune des trois.

```
1: cont_egal r1, 0, 5
2: cont_egal r2, 0, 5
3: charge r3, 0
4: continue 6
5: somme r3 r1 r2
6: // fin (stop)
```