ICC MT & EL Test final partie théorique /White vendredi 20 décembre 2019

Ne PAS retourner ces feuilles avant d'en être autorisé!

Merci de poser votre carte CAMIPRO en évidence sur la table.

Vous pouvez déjà compléter et lire les informations ci-dessous:

NOM _____

Prénom _____

Numéro SCIPER _____

Signature _____

BROUILLON : Ecrivez aussi votre NOM-Prénom sur la feuille de brouillon fournie. Toutes vos réponses doivent être sur cette copie d'examen. Les feuilles de brouillon sont ramassées pour être immédiatement détruites.

Le test écrit commence à: **14h15**

Nous recommandons de finir cet examen à : **15h30**

Pour commencer l'examen de C++ qui se terminera à : **16h40**

Le contrôle final de ICC reste SANS appareil électronique

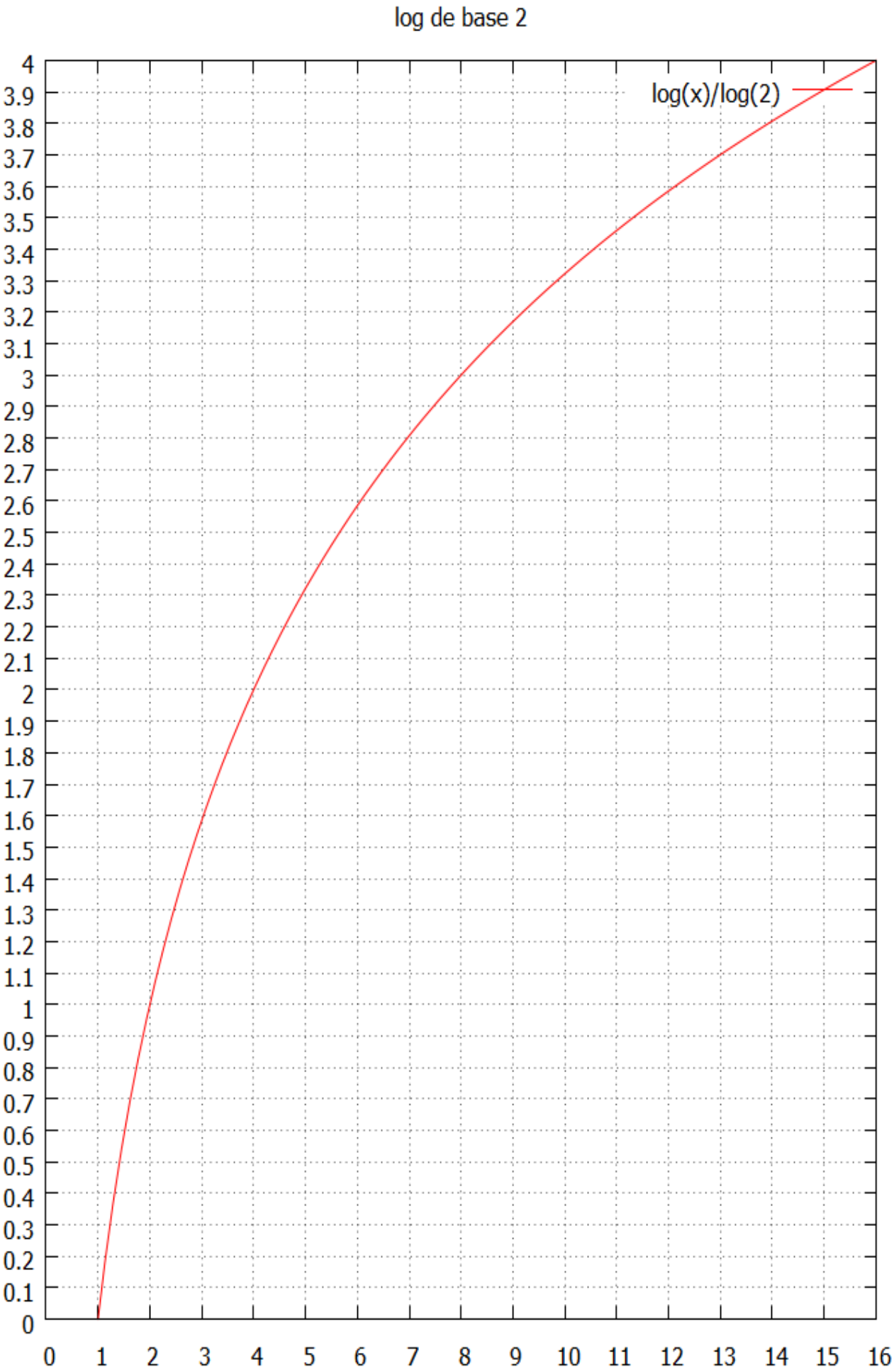
Vous avez le droit d'avoir tous vos documents **personnels** sous forme papier: dictionnaire, livres, cours, exercices, code, projet, etc...

Total sur 25 points = 17 points pour la partie Quizz et 8 points pour les questions ouvertes

Vous pouvez utiliser un crayon à papier et une gomme

La partie Quizz (QCM) comporte 12 questions : chaque question n'a qu'une seule réponse correcte parmi les 4 réponses proposées. Chaque réponse correcte donne 1 point (par défaut) ou 2 points (fond noir ci-dessous). Aucun point n'est donné en cas de réponses multiples, de rature, ou de réponse incorrecte. ***Indiquez vos réponses à la partie Quizz dans ce tableau :***

	Questions du Quizz												
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	
A													**A**
B													**B**
C													**C**
D													**D**



Une précision du centième est parfois nécessaire pour vos calculs ; souvent le dixième suffit.

Unités de taille utilisées dans cet examen

Les puissances de dix utilisent une seule lettre majuscule comme par exemple :
10^3 = K (kilo), 10^6 = M (méga), 10^9 = G (giga), 10^{12} = T (téra)

Dans cet examen, nous utilisons la lettre majuscule '**B**', comme ***Byte***, pour désigner un ***octet***.
La lettre minuscule '**b**' désigne un **bit**.

Opérateurs logiques

and (et logique), or (ou inclusif), xor (ou exclusif), not (négation logique)

Formules trigonométriques

```
sin(u)sin(v) = 0.5(cos(u - v) - cos(u + v))
cos(u)sin(v) = 0.5(sin(u + v) - sin(u - v))
```

Syntaxe des instructions assembleur

Attention : certaines questions utilisent seulement un sous-ensemble de ces instructions

Chargement d'un registre : Il est possible de remplacer **r2** par une constante :

`charge r1, r2`	range dans **r1** la valeur stockée dans **r2**
`mcharge r1, r2`	range dans **r1** la valeur stockée en mémoire à l'adresse **r2**

Calcul : Il est possible de remplacer **r3** par une constante :

`somme r1, r2, r3`	calcule **r2 + r3** et range le résultat dans **r1**
`soustrait r1, r2, r3`	calcule **r2 - r3** et range le résultat dans **r1**
`multiplie r1, r2, r3`	calcule **r2 * r3** et range le résultat dans **r1**
`divise r1, r2, r3`	calcule **r2/r3** et range le résultat dans **r1**

Branchement : Il est possible de remplacer **r2** par une constante :

`continue_diff r1, r2, i`	Si **r1** est différent de **r2** alors continue à l'adresse **i** Sinon passe à l'instruction suivante.
`continue_egal r1, r2, i`	Si **r1** est égal à **r2** alors continue à l'adresse **i** Sinon passe à l'instruction suivante.
`continue_ppe r1, r2, i`	Si **r1** $\leq$ **r2** alors continue à l'adresse **i** Sinon passe à l'instruction suivante.
`continue_pp r1, r2, i`	Si **r1** $<$ **r2** alors continue à l'adresse **i** Sinon passe à l'instruction suivante.
`continue_pge r1, r2, i`	Si **r1** $\geq$ **r2** alors continue à l'adresse **i** Sinon passe à l'instruction suivante.
`continue_pg r1, r2, i`	Si **r1** $>$ **r2** alors continue à l'adresse **i** Sinon passe à l'instruction suivante.
`continue i`	branchement inconditionnel à l'adresse **i**

Divers : arrêt et fin du programme avec l'instruction **stop**

QUIZZ ICC Théorie modules 2 et 3

Question 1 : lequel des mots suivants possède la plus grande entropie?

A CREMIERE
B CHOCOLAT
C HOPITAUX
D GLACIERE

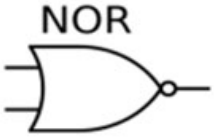
Question 2 : savoir utiliser les bonnes informations.
Soit une mémoire flash USB pouvant stocker jusqu´à 25GB. Sur cette mémoire se trouve le fichier d´une vidéo de 20GB. Cette video est de résolution 1080x1920 pixels pour 24fps (images par seconde) et dure 1h40. Vous souhaitez convertir la video afin de pouvoir la stocker et la lire sur votre ordinateur portable dernier cri offrant une résolution de 3840 × 2160 avec un taux de rafraichissement de 144Hz.
Toujours à la pointe de la technologie, vous utilisez le format appelé *webm* qui est utilisé par Google pour stocker les vidéos sur la plateforme de contenu Youtube. La vitesse d'encodage dans le format *webm* est de **x0.1** (cela signifie que pour une minute de video l'encodage dure 10 minutes).

Après 2h d'encodage le fichier (partiel) produit par l'encodage a déjà une taille de 120Mo.
Quel espace mémoire *une* image video occupe-t-elle dans le fichier produit par l'encodage ?
Voici quelques valeurs pouvant éventuellement être utiles pour vos calculs:
100/24 ~ 4.17
100/144 ~ 0.694
100/864 ~ 0.116
200/144 ~ 1.388
250/144 ~ 1.736
1000/1728 ~ 0.578

A 6.94 KB
B 13.88KB
C 417. KB
D 1.16 KB

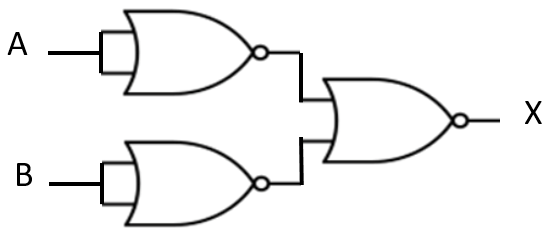
Question 3 (2 pts) : A quelle porte logique le circuit mystère est-il équivalent ?

Porte logique NOR:



Inputs		Output
A	B	F
0	0	1
1	0	0
0	1	0
1	1	0

Circuit mystère construit avec des portes NOR :



Le circuit mystère est équivalent à la porte logique …
A XOR
B AND
C NAND
D OR

Question 4 : Assembleur1

Il existe différentes méthodes pour déterminer le nombre Pi. Nous allons utiliser le développement de Taylor de la fonction arctan, évalué au point 1 appelé aussi formule de Leibniz-Gregory puis multiplier 4 afin d'obtenir une approximation du nombre Pi.

$$4\sum_{k=0}^{n}\frac{(-1)^k}{2k+1} = \frac{4}{1} - \frac{4}{3} + \frac{4}{5} - \frac{4}{7} \ldots . \approx \pi$$

Afin d'obtenir une bonne précision de Pi à 10^{-2}, on calcule 200 termes de l'approximation.

Voir la page 3 pour les conventions utilisées pour les instructions.
Pour cet exercice, l'instruction **`divise`** produit un *nombre à virgule*.

```
1 : charge        r1, 200
2 : charge        r2, 0
3 : charge        r3, 0
4 : charge        r4,-1
5 : continue_pg   r2, r1, _①__
6 : multiplie     r4, r4, -1
7 : charge        r5, 0
8 : charge        r6, 0
9 : multiplie     r6, r2, 2
10: somme         r6, r6, 1
11: divise        r5, __②__, r6
12: somme         r3, r3, r5
13: somme         r2, r2, 1
14: continue      5
15: multiplie     r3, r3, 4
16: stop
```

L'algorithme est incomplet ; par quoi remplacez-vous les espaces vides ① et ② ?

A ① = 6 et ② = r3

B ① = 15 et ② = r4

C ① = 6 et ② = r4

D ① = 15 et ② = r3

Question 5 :

La mémoire cache peut contenir 3 blocs de 2 mots. Si on utilise la solution de remplacement de cache LRU vue en cours, combien de défauts de cache génère la séquence d'accès mémoire suivante (adresses de mots) en partant d'une mémoire cache vide :

11 12 5 10 11 13 7 11 8 6 12

Les adresses commencent à 0 et les blocs sont toujours alignés sur les multiples de 2 en mémoire centrale.

A 6
B 5
C 7
D 8

Question 6 (2 pts) : Assembleur2
Voir la page 3 pour les conventions utilisées pour les instructions.
Pour cet exercice, l'instruction **`divise`** est la *division entière*.

```
1 : charge          r1,n
2 : charge          r2,0
3 : charge          r3,0
4 : continue_pp     r1,2,__ ①__
5 : divise          r4,r1,2
6 : multiplie       r3,r4,2
7 : continue_egal   r3,r1,13
8 : multiplie       r3,r1,3
9 : somme           r1,r3,1
10: somme           r2,r2,1
11: continue_diff   r1,1, 5
12: continue        15
13: divise          r1,r1,2
14: continue        10
15: charge          r5,__②__
16: stop
```

Selon Wikipedia, en mathématiques, on appelle suite de Syracuse une suite d'entiers naturels définie de la manière suivante : on part d'un nombre entier **n** plus grand que zéro ; s'il est pair, on le divise par 2 ; s'il est impair, on le multiplie par 3 et on ajoute 1. Ensuite on répète l'opération à partir du résultat obtenu à l'étape précédente ; on obtient ainsi une suite d'entiers positifs.

Le code assembleur ci-dessus effectue ces opérations à partir de la valeur **n** qui est chargée dans **r1**. De plus le programme incrémente un compteur à chaque calcul d'une valeur de la suite. Cela est fait jusqu'au moment où la valeur de la suite vaut **1** ; dans ce cas on quitte le programme en chargeant la valeur du compteur dans le registre **r5.**

L'algorithme est incomplet ; par quoi remplacez-vous les espaces vides ① et ② ?

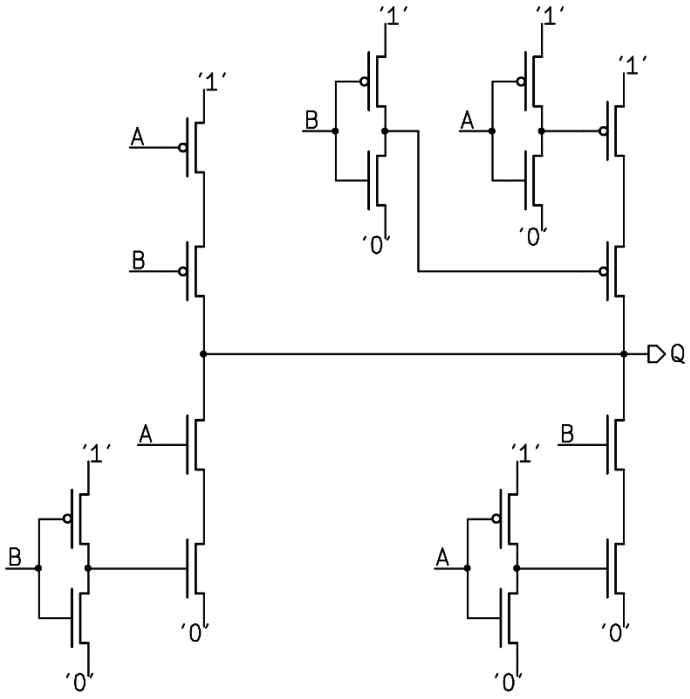
A ① = 13 et ② = r1
B ① = 13 et ② = r2
C ① = 15 et ② = r2
D ① = 8 et ② = r2

Question 7: Soit $X(t) = \sin(2\pi f_1 t) + (\sin(2\pi f_2 t))^2$, avec $f_1 > 2f_2 > 0$.

Quelle est la bande passante du signal X(t) ?

A $2f_2$
B $2(f_1 + f_2)$
C $f_1 + 2f_2$
D f_1

Question 8 (2 pts): Quelle est la table de vérité de la sortie Q du circuit ci-dessous ?



A	B	X
0	0	1
0	1	1
1	0	1
1	1	0

A	B	X
0	0	1
0	1	0
1	0	0
1	1	1

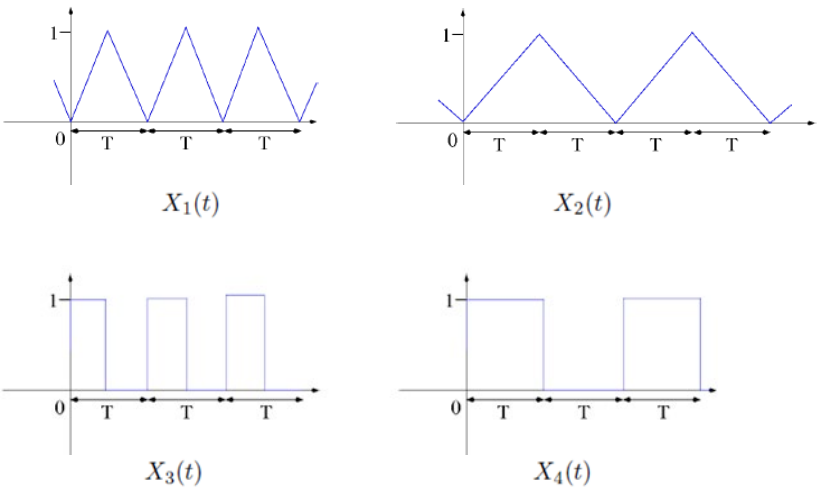
A	B	X
0	0	0
0	1	1
1	0	1
1	1	0

A	B	X
0	0	1
0	1	0
1	0	0
1	1	0

Réponses : A B C D

Question 9 (2 pts): On utilise un filtre à moyenne mobile de période d'intégration $T_c > 0$ produisant un signal filtré $\hat{X}(t) = \frac{1}{T_c}\int_{t-T_c}^{t} X(s)ds$.

Quelle proposition est vraie à propos des 4 signaux ci-dessous avec $T = T_c$



A $\hat{X}_1(t) = \hat{X}_3(t) = \hat{X}_4(t) \neq \hat{X}_2(t) = X_4(t)$

B $\hat{X}_1(t) = \hat{X}_3(t) \neq \hat{X}_2(t) \neq \hat{X}_4(t) = X_2(t) \neq \hat{X}_1(t)$

C $\hat{X}_1(t) = \hat{X}_2(t) = \hat{X}_3(t) = \hat{X}_4(t)$

D $\hat{X}_1(t) \neq \hat{X}_2(t) \neq \hat{X}_3(t) = \hat{X}_1(t) = \hat{X}_4(t)$

Question 10 : Entropie, codage et performances

Un message X est composée des 9 lettres suivantes { U, P, F, L, I, S, Z, E, N} avec les probabilités correspondantes pour chaque lettre {1/4 , 1/2, 1/16, 1/128, 1/8, 1/32, 1/256, 1/64, 1/256}.
On note H(X) l'entropie du message X, $L_{SF}(X)$ la performance du code obtenu avec Shannon-Fano et $L_H(X)$ la performance du code obtenu avec Hufman.

Quelle proposition est vraie ?

A $H(X) = L_H(X) < L_{SF}(X)$
B $H(X) = L_H(X) = L_{SF}(X)$
C $H(X) < L_H(X) = L_{SF}(X)$
D $H(X) < L_H(X) < L_{SF}(X)$

Question 11 (2 pts): Le message "**yahoo tatoo taboos**" (sans les espaces) est encodé par l'algorithme de Huffman. Quel est le nombre total de bits obtenu pour le codage de cette séquence ?

A 40
B 41
C 42
D 39

Question 12 : échantillonnage
On dispose d'un dispositif de type Convertisseur-Analogique-Numérique (CAN) qui échantillonne un signal audio avec une période T_e compatible avec la fréquence utile f_u du signal. Malheureusement, à la fréquence utile f_u s'ajoute un bruit parasite à haute fréquence f_b qui produit une distorsion lors de la reconstruction à partir des échantillons. Quel type de filtrage faut-il faire pour éviter cette distorsion ?

A Filtrer le signal *après* l'étape d'échantillonnage avec un filtre passe-bas idéal de fréquence de coupure f_c légèrement inférieure à $1/(2T_e)$

B Filtrer le signal *avant* l'étape d'échantillonnage avec un filtre passe-bas idéal de fréquence de coupure *fc* légèrement inférieure à $1/(2T_e)$

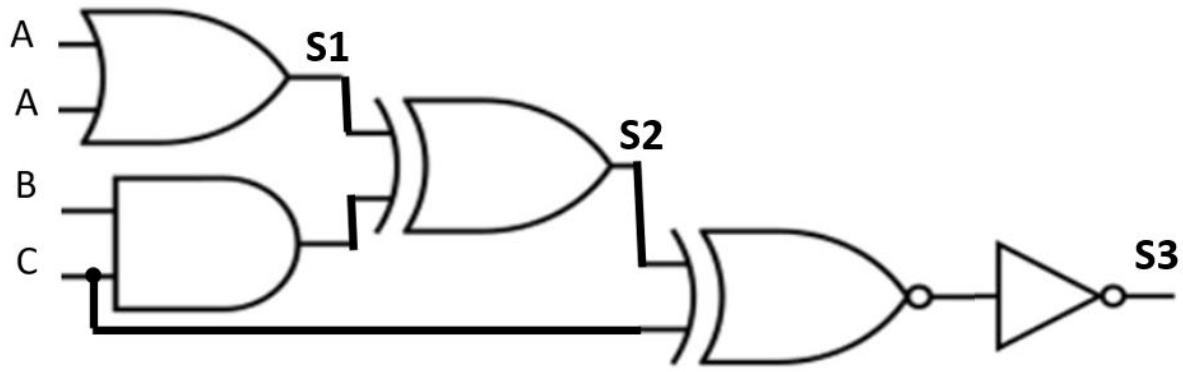
C Filtrer le signal *avant* l'étape d'échantillonnage avec un filtre passe-bas idéal de fréquence de coupure *fc* légèrement inférieure à $2/T_e$

D Filtrer le signal *avant* l'étape d'échantillonnage avec un filtre à moyenne mobile de période d'intégration légèrement supérieure à $2T_e$

Questions Ouvertes

Question 1 : (4 pts) circuit à base de portes logiques

Soit le circuit suivant construit à partir des portes logiques vues en exercices (M3.L1) :



1.1) déterminer la table de vérité de la sortie intermédiaire S1 en fonction de l'entrée A

A	S1

1.2) en déduire la table de vérité de la sortie intermédiaire S2 en fonction de S1 , B et C

S1	B	C	S2

1.3) finalement, déterminer la table de vérité du circuit (sortie S3)

1.3.1) en fonction de S2 et C

S2	C	S3

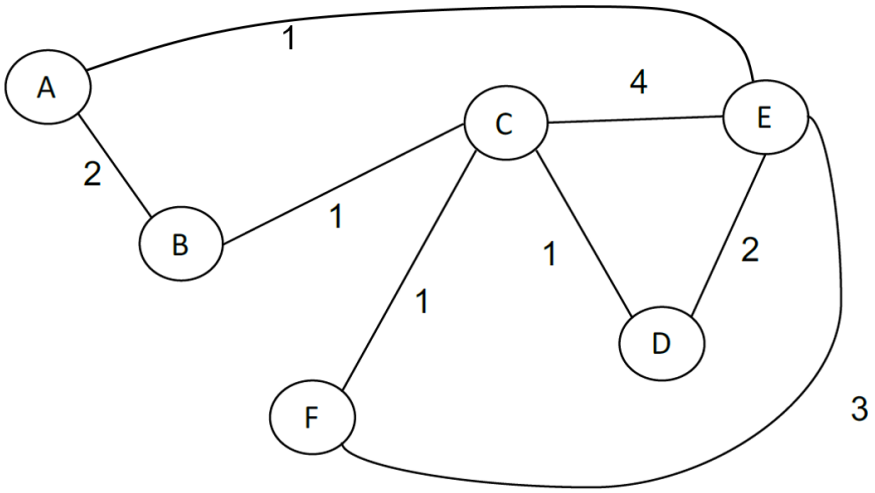
1.3.2) en fonction de A, B et C

A	B	C	S3

Question 2 : (4 pts) Tables de routage

2.1) Construire les tables de routage de chacun des nœuds du réseau suivant. La table doit indiquer, pour chaque *Destination*, le nom du premier noeud du chemin (*Direction*) et la *Distance* exprimées par le coût total jusqu'à la destination. Le coût est la quantité indiquée à coté de chaque lien sur le dessin ; par exemple le coût entre D et E est 2.

A coût égal il faut choisir la direction avec le moins de nœuds intermédiaires.



A.

Dest.	Dir.	Dist.
B		
C		
D		
E		
F		

B.

Dest.	Dir.	Dist.
A		
C		
D		
E		
F		

C.

Dest.	Dir.	Dist.
A		
B		
D		
E		
F		

D.

Dest.	Dir.	Dist.
A		
B		
C		
E		
F		

E.

Dest.	Dir.	Dist.
A		
B		
C		
D		
F		

F.

Dest.	Dir.	Dist.
A		
B		
C		
D		
E		

2.2) Les **liens** directs **E <-> D** et **B <-> C** sont coupés à cause d'un ouragan… Ecrire seulement la liste des **lignes qui changent** dans les tables des nœuds **B** et **D**, sous la forme suivante :

Dans la table du nœud B :
Destination, Direction, Distance

Dans la table du nœud D :
Destination, Direction, Distance

Rappel : avez-vous complété le tableau en page 1 ?

Masquez cet examen pendant que vous travaillez sur l'examen de C++