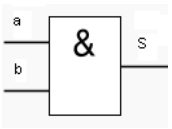
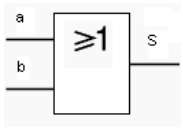
Ds - Fcts logiques – Routage – Piles/Files

1- EXERCICE 1 : FONCTIONS LOGIQUES





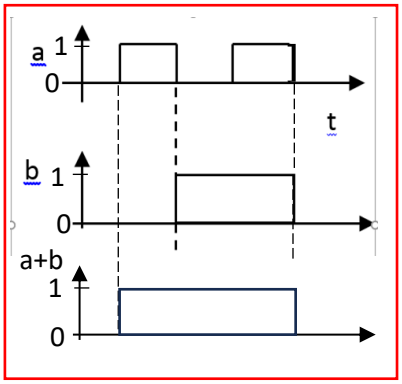
On donne ci-contre, les symboles logiques des fonctions ET et OU :

1- Compléter sur feuille de copie la table de vérité de cette fonction.

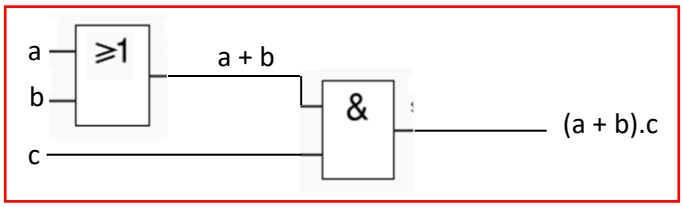
a	b	a + b
0	0	0
0	1	1
1	0	1
1	1	1

2- On donne ci-contre, le chronogramme des variables booléennes a et b.

Tracer sur feuille de copie ces chronogrammes avec en plus celui de la variable (a+b) en de-dessous, dans l'alignement.



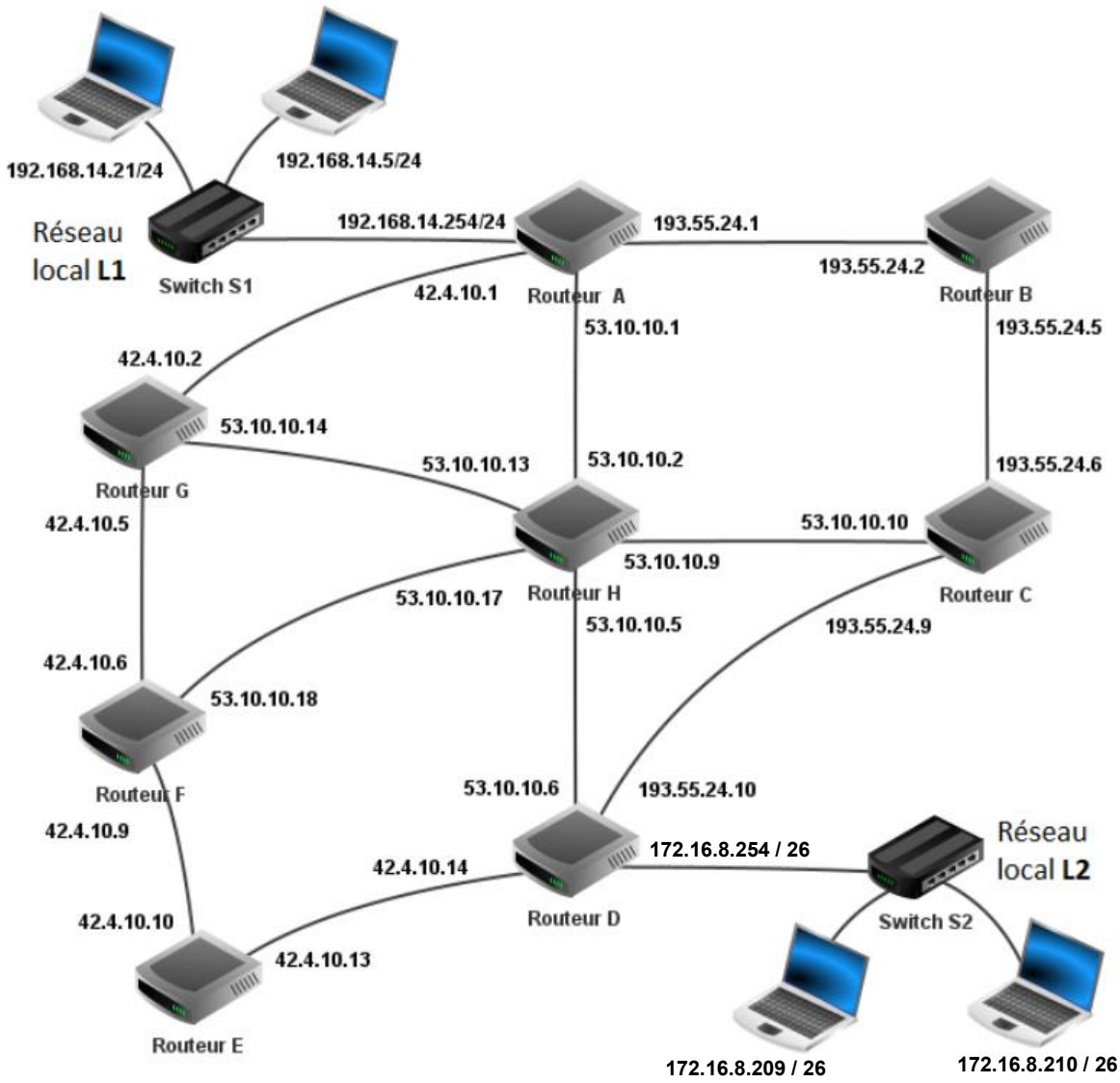
3- Tracer sur feuille de copie le logigramme de $S = (a + b).c$



4- Simplifier les expressions $a.\bar{a}$ et $a + \bar{a}$: $a.\bar{a} = 0$ et $a + \bar{a} = 1$

2- EXERCICE 2 : ROUTAGE

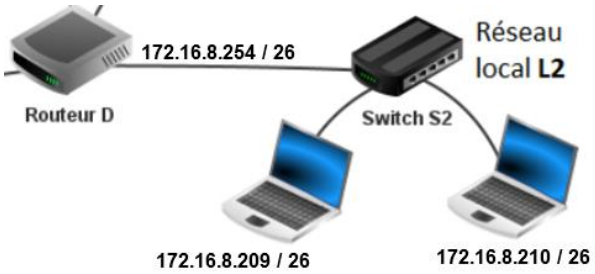
On considère le réseau dont le plan est donné ci-dessous :



Partie A : Adresses IP

1- Concernant le réseau local L2 :

a. Donner l'adresse Ip 172.16.8.209 en écriture binaire.



172	16	8	209
1010 1100	0001 0000	0000 1000	1101 0001

b. Sur cette adresse, différencier les parties Réseau et Hôte en les encadrant.

L'adresse Ip en binaire est donc :

1010 1100 . 0001 0000 . 0000 1000 . 1101 0001

La partie réseau est encadrée à gauche. Elle correspond aux 26 premiers bits. La partie hôte est encadrée à droite, elle correspond aux 6 bits restant.

c. Donner le masque de sous réseau en notation binaire, puis décimale.

Le masque du réseau en binaire est : 11111111 . 11111111 . 11111111 . 11000000

En décimal, il devient : 255.255.255.192

d. Donner l'adresse du réseau en écriture binaire, puis en décimale.

L'adresse du réseau en binaire est : 1010 1100 . 0001 0000 . 0000 1000 . 1100 0000

En décimal, cette adresse devient : 172.16.8.192

e. Donner l'adresse de diffusion en écriture binaire, puis en décimale.

L'adresse de diffusion en binaire est : 1010 1100 . 0001 0000 . 0000 1000 . 1111 1111

En décimal, cette adresse devient : 172.16.8.255

f. Donner le nombre maximum de machines pouvant être connectées à ce réseau.

La partie hôte est composée de 6 bits. Le nombre maximum de machines que peut contenir ce réseau est donc de $2^6 - 2 = 62$ machines.

g. Un ordinateur d'adresse Ip 172.16.8.128 / 26 est relié au Switch du réseau L2 . Sur un terminal de cet ordinateur, on exécute ping 172.16.8.209 . Que se passe-t-il ?

Le dernier octet de l'adresse, 128, correspond en binaire au nombre 1000 0000. La partie réseau de cette adresse IP n'est donc pas égale à celle du réseau L2 car le dernier octet ne commence pas par les bits 11. Ainsi la communication entre cet ordinateur et les autres du réseau L2 sera impossible et donc le ping échouera.

Partie B : Protocoles de routage

On donne ci-dessous des extraits des tables de routage des routeurs :

Routeur	Réseau destinataire	Passerelle	Interface
A	L2	53.10.10.2	53.10.10.1
B	L2	193.55.24.6	193.55.24.5
C	L2	193.55.24.10	193.55.24.9
D	L2	Connecté	172.16.8.254
E	L2	42.4.10.14	42.4.10.13
F	L2	42.4.10.10	42.4.10.9
G	L2	53.10.10.13	53.10.10.14
H	L2	53.10.10.6	53.10.10.5

2- À l'aide des extraits des tables de routage ci-dessus, donner un chemin (c'est-à-dire nommer les routeurs traversés) suivi par un message envoyé du réseau L1 vers le réseau L2.

L1 – **A** - 53.10.10.2 de **H** – 53.10.10.6 de **D** – L2

La liaison entre les routeurs H et D est rompue :

3- Sachant que le protocole de routage RIP est utilisé, donner les 2 nouveaux chemins que pourra suivre un message allant de L1 vers L2.

Chemin 1 : L1 – **A** - 53.10.10.2 de **H** – 53.10.10.10 de **C** – 193.55.24.10 de **D** – L2

Chemin 2 : L1 – **A** - 193.55.24.2 de **B** –193.55.24.6 de **C** – 193.55.24.10 de **D** – L2

4- Choisir un des chemins de la question précédente. Donner les routeurs dont la règle de routage à destination de L2 est obligatoirement modifiée. Ecrire sur votre copie les lignes qui sont à modifier.

Pour le chemin 1, on modifie la table de routage de H , pour le réseau L2 :

- Passerelle : 53.10.10.10
- Interface : 53.10.10.9

La liaison entre les routeurs H et D est rétablie. Pour tenir compte du débit des liaisons, on décide d'utiliser le protocole OSPF pour effectuer le routage. Le coût d'une liaison est donné ici par la formule : $coût = \frac{10^9}{BP}$ où BP est la bande passante de la connexion en bit par seconde. Les valeurs des bandes passantes de chaque liaison entre les routeurs sont données ci-dessous :

Liaison	Bande passante
A-B	1 Gbit/s
A-H	1 Gbit/s
A-G	1 Gbit/s
B-C	1 Gbit/s
C-H	100 Mbit/s
C-D	1 Gbit/s

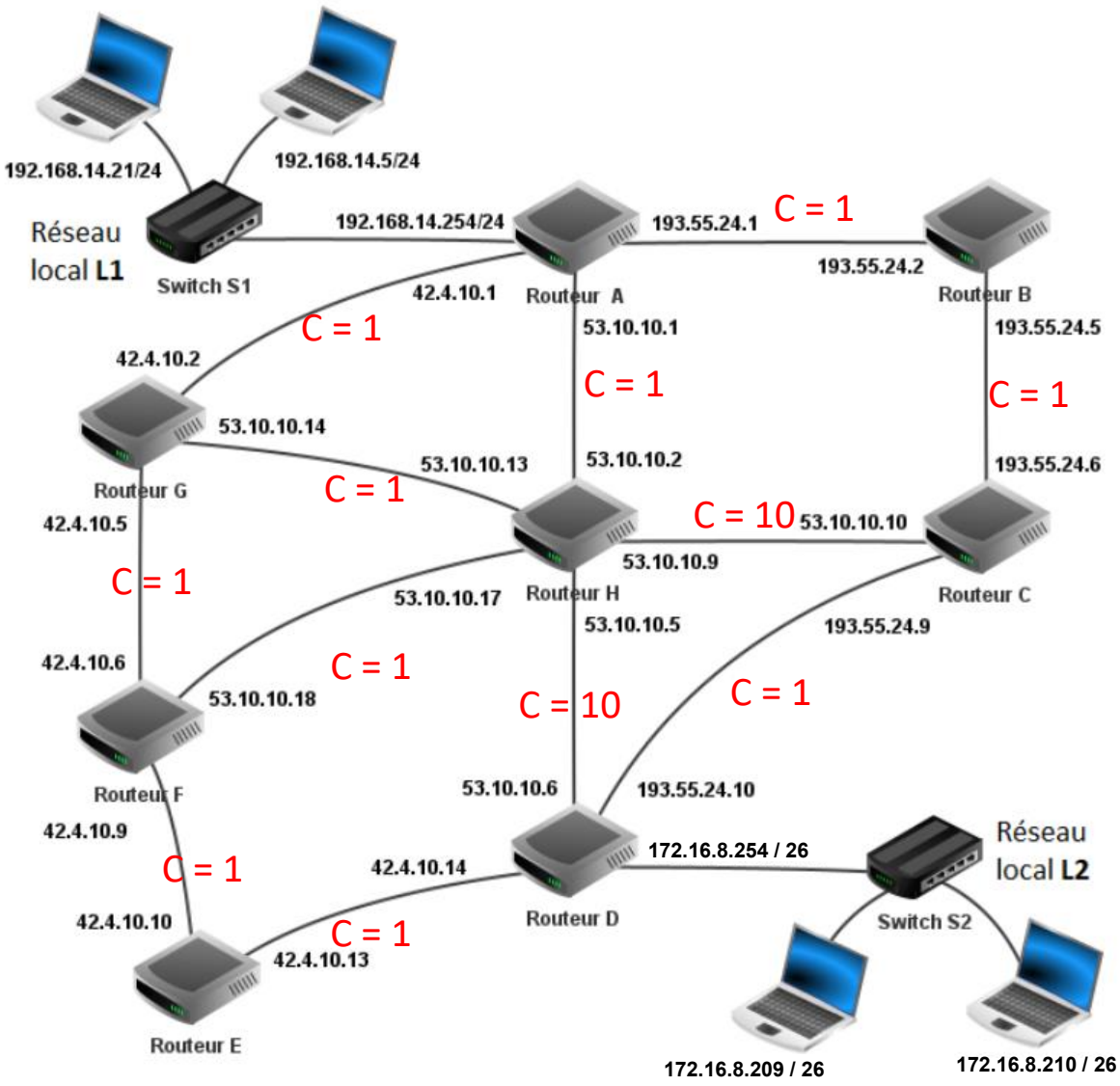
Liaison	Bande passante
D-H	100 Mbit/s
D-E	10 Gbit/s
E-F	10 Gbit/s
F-H	1 Gbit/s
F-G	10 Gbit/s
G-H	1 Gbit/s

5- Calculer le coût des liaisons pour les 3 valeurs de bande passante qui apparaissent dans le tableau ci-dessus.

Les couts sont :

- 100 Mbits/s : $c = \frac{10^9}{100\ 10^6} = \frac{10^9}{10^8} = 10$
- 1 Gbits/s : $c = \frac{10^9}{10^9} = 1$
- 10 Gbits/s : $c = \frac{10^9}{10\ 10^9} = \frac{10^9}{10^{10}} = 0{,}1$. Le coût est fixé à 1 car le protocole OSPF n'accepte pas des coûts inférieurs à 1.

6- Déterminer alors le chemin que suivra un message allant de L1 vers L2 et donner son coût.



Le chemin sera : L1 – **A** – **B** – **C** – **D** – L2 . Son coût sera de 3 .

3- EXERCICE 3 : PILE

1- On donne ci-contre le code incomplet de la classe Pile.

Ecrire sur feuille de copie le code des méthodes empiler(), estVide(), depiler() et taille().

CORRIGE

```
class Pile:
    def __init__(self,nom=''):
        self.nom = nom
        self.contenu = []

    def empiler(self, valeur):
        self.contenu.append(valeur)

    def estVide(self):
        return self.contenu == []

    def depiler(self):
        if not self.estVide() :
            return self.contenu.pop()

    def taille(self):
        return len(self.contenu)
```

2- En exécutant le programme principal ci-contre, on obtient dans la console :

```
Etat de la pile p :

| 7  |
| 25 |
| 12 |
| 9  |
| 5  |
| 1  |
------
le maximun de la pile p est 25

Etat de la pile p :

| 7  |
| 25 |
| 12 |
| 9  |
| 5  |
| 1  |
------
```

```
# Main
if __name__ == '__main__' :
    p = Pile('p')
    tab = [1,5,9,12,25,7]
    for nb in tab :
        p.empiler(nb)
    print(p)
    m = p.maxi()
    print(f"le maximun de la pile {p.nom} est {m}")
    print(p)
```

Ainsi la méthode maxi() renvoie le maximum des nombres empilés dans la pile.

Ecrire sur feuille de copie le script de cette méthode sans utiliser directement la liste de l'attribut contenu, mais en réalisant un parcours de la pile avec les méthodes empiler(), depiler() et estVide() ou taille(). Utiliser une pile auxiliaire afin de ne pas modifier le contenu de la pile après exécution de cette méthode.

CORRIGE

```
def maxi(self) :
    q = Pile()
    premier = True
    while not p.estVide() :
        n = self.depiler()
        if premier == True or n > max :
            max = n
            premier = False
        q.empiler(n)
    while not q.estVide() :
        n = q.depiler()
        self.empiler(n)
    return max
```