Métropole – 2026 – sujet2 - Correction

Exercice 1 (6 points)

Partie A – Adressage et routage

1. Adresse du réseau du site C

Le suffixe /16 conserve les 16 premiers bits, c'est-à-dire les deux premiers octets : l'adresse du réseau est **172.16.0.0**.

2. Adresse de diffusion

On met à 1 les 16 bits de la partie machine : **172.16.255.255**.

3. Nombre maximal de machines

16 bits pour les machines : 2^{16} = 65 536 adresses, moins les 2 adresses réservées : **65 534 machines**.

4. Table de routage de R1 (RIP)

Destination	Passe par	Nombre de sauts
R2	**R3**	**2**
R3	**R3**	**1**
R4	**R4**	**1**
R5	**R4**	**2**

👉 *Commentaire : d'après la figure, R1 est relié à R3 et R4 ; R2 est relié à R3 et R5 ; R4 à R1, R3 et R5. R2 est à 2 sauts par R3 (par R4 et R5, il en faudrait 3).*

5. Chemin de PC A01 à PC B01 (RIP)

PC A01 → switch A → R1 → R3 → R2 → switch B → PC B01.

6. Panne de R3

R1 ne peut plus passer par R3 : la route devient **R1 → R4 → R5 → R2** (3 sauts).

7. Coût OSPF des liaisons

Type de connexion	Débit (bit/s)	Coût
Ethernet	10^7	**10**
Fast Ethernet	10^8	**1**
Fibre	10^9	**0,1**

8. Chemin OSPF de R1 à R4

Liaison directe R1 – R4 (Ethernet) : coût **10**. Par R3 : R1 – R3 (fibre, 0,1) puis R3 – R4 (Fast Ethernet, 1) : coût **1,1**.

OSPF choisit le chemin de coût minimal : **R1 → R3 → R4**.

👉 *Commentaire : le chemin R1 – R3 – R2 – R5 – R4 coûterait 0,1 + 10 + 1 + 1 = 12,1.*

Partie B – Programmation

9. La contrainte n'est pas vérifiée

ajouter_route ajoute la route **sans jamais comparer** la longueur de liste_routes à MAX_ROUTES : on peut appeler la fonction autant de fois qu'on veut, la liste dépassera 5 routes. MAX_ROUTES n'est utilisée nulle part.

10. Classe Routage (lignes 3, 7, 8 et 10)

```
class Routage:
    def __init__(self, capacite = 5):
        self.capacite = capacite                    # ligne 3
        self.routes = []

    def ajouter(self, route):
        if len(self.routes) < self.capacite:        # ligne 7
            self.routes.append(route)               # ligne 8
        else:
            print('Erreur : capacité maximale atteinte')   # ligne 10
```

👉 *Commentaire : capacite = 5 dans l'en-tête donne la valeur par défaut : Routage() crée un objet de capacité 5, Routage(10) de capacité 10.*

11. Méthode afficher

```
    def afficher(self):
        for route in self.routes:
            for routeur in route:
                print(routeur)
```

```
print('---')
```