

Exercice 1 :

Partie A : Réseau dans un lycée

1. Il reste un octet soit 8 bits pour l'adresse réseau (1^{re} adresse), les machines et l'adresse de broadcast (dernière adresse). Ainsi le réseau peut accueillir : $2^8 - 2 = 254$ machines.

2. Conversion décimal – binaire : $217_{(10)} = 128 + 64 + 16 + 8 + 1 = 1101\ 1001_{(2)}$

3. Conversion binaire – décimal : $110010_{(2)} = 32 + 16 + 2 = 50_{(10)}$

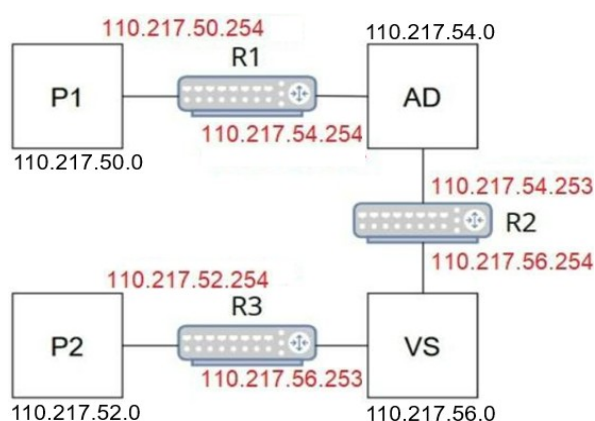
4. L'adresse du sous-réseau de la machine 110.217.53.22/24 est 110.217.53.0 donc cette machine n'appartient pas au réseau pédagogique 2.

[Le réseau pédagogique 2 (110.217.52.0/24) admet une plage d'adresse de 110.217.52.0 à 110.217.52.255.]

5. Report des adresses sur le schéma du réseau :

Table de routage du routeur R1 :

Destination	Passerelle	Interface
110.217.50.0	on-link	110.217.50.254
110.217.52.0	110.217.54.253	110.217.54.254
110.217.54.0	on-link	110.217.54.254
110.217.56.0	110.217.54.253	110.217.54.254



6. Table de routage du routeur R1 :

Destination	Passerelle	Interface
110.217.50.0	on-link	110.217.50.254
110.217.52.0	110.217.50.253	110.217.50.254
110.217.54.0	on-link	110.217.54.254
110.217.56.0	110.217.54.253	110.217.54.254

7. Le nombre de saut étant identique, aucune modification n'est nécessaire dans la table de routage du routeur R2.

Partie B : Réseaux et graphes

8. Qu'il y ait ou non un chemin reliant les deux sommets, la fonction risque d'empiler les appels récursifs jusqu'à atteindre la limite de récursion car il n'y a pas de mémoire des sommets déjà visités.

Exemple d'empilement d'appel sans fin pour une graphe avec au moins deux sommets A et B liés entre eux :

```
recherche("A", "E")  
appelle recherche("B", "E")  
qui appelle recherche("A", "E")  
et ainsi de suite ...
```

8. Proposition de solution du problème :

```
def recherche_v2(R1, R2, deja_vu=None):  
    if deja_vu is None:  
        deja_vu = []  
  
    if R1 == R2:  
        return True  
  
    if R1 not in deja_vu:  
        deja_vu.append(R1)  
  
    for S in adjacents(R1,G):  
        if S not in deja_vu:  
            if recherche_v2(S, R2, deja_vu):  
                return True  
  
    return False
```