# Adressage réseau

## Paire réseaux / masque : notation CIDR

On rappelle qu'une adresse IP version 4 est de la forme xxx.xxx.xxx.xxx, soit 4 octets, généralement écrits en base 10. On ne parlera pas d'IP version 6.

Un réseau possède une adresse IP, chaque machine du réseau également, et une dernière adresse IP, la « plus grande » dite de *broadcast*, est réservée à la diffusion sur tout le réseau.

La notation CIDR permet de donner l'adresse et la taille d'un réseau. Elle est de la forme

xxx.xxx.xxx.xxx / n

Où n est un entier compris entre 1 et 32 et le reste une adresse IP (celle du réseau).
Le nombre n correspond au nombre de bits fixes en partant de la gauche, les autres bits sont libres.
Ainsi plus n est petit, plus il y a de bits libres, plus il y a d'adresses disponibles sur le réseau et plus celui-ci est susceptible d'être grand.

## Exemple

En notation CIDR, mon réseau domestique est 192.168.1.0 / 24.

- 192.168.1.0 est l'adresse du réseau ;
- les 24 bits en partant de la gauche sont fixes, donc les 3 premiers octets sont fixes, et ainsi seul le dernier octet de l'IP peut varier ;
- les adresses IP du réseau sont de la forme 192.168.1.xxx ;
- l'adresse de *broadcast* est 192.168.1.255.

## Définition : masque de sous-réseau

Le masque de sous réseau d'un réseau de la forme xxx.xxx.xxx.xxx / n consiste en une adresse IP dont les n premiers bits sont à 1 et les autres à 0.

**Exemple**

Le masque de sous-réseau de 192.168.1.0 / 24 est

```
11111111 . 11111111 . 11111111 . 00000000
```

**Méthode : trouver l'adresse d'un réseau**

L'IP de ma machine dans un réseau xxx.xxx.xxx.xxx / 20 est 192.168.181.3.

Quelle est l'adresse du réseau ?

- On écrit l'IP de la machine en binaire, on obtient

```
11000000 . 10101000 . 10110101 . 00000011
```

- Le masque de sous réseau est

```
11111111 . 11111111 . 11110000 . 00000000
```

- On fait un ET bit à bit entre ces deux adresses, on obtient

```
11000000 . 10101000 . 10110000 . 00000000
```

- On écrit cela en décimal : l'adresse IP du réseau est 192.168.176.0.

**Méthode : trouver l'adresse de diffusion**

Quelle est l'adresse de diffusion du réseau précédent ?

- On reprend l'adresse du réseau en binaire

```
11000000 . 10101000 . 10110000 . 00000000
```

- Puisque le réseau est en / 20, on met les 12 derniers bits à 1 :

```
11000000 . 10101000 . 10111111 . 11111111
```

- En écrivant en décimal, l'adresse IP de diffusion sur le réseau est 192.168.191.255.

## Exercice 1 : réseaux et masques

Dans chaque cas, l'IP d'une machine sur un réseau est donnée, ainsi que le masque du réseau.

- Dire combien d'IP comporte le réseau et combien de machines peuvent être adressées.
- Retrouver l'IP du réseau.
- Donner l'IP de *broadcast*.

1. 202.2.18.149 sur un réseau en / 8.
2. 97.124.36.142 sur un réseau en / 24.
3. 192.168.180.57 sur un réseau en / 18.

## Exercice 2

Dire si l'IP appartient au réseau.

1. 172.26.21.46 et 172.26.21.0 / 25
2. 192.168.186.240 et 192.168.186.224 / 29
3. 172.18.47.54 et 172.18.46.0 / 23

## Exercice 3 : projet qu'on fera sans doute après les écrits

Les méthodes vues précédemment peuvent être automatisées avec un programme.

Tu peux écrire une classe `IP` pour représenter une IP, implémenter la méthode `__str__` et également la méthode `__and__` pour effectuer un `et` bit à bit entre 2 IPs, ce qui te permettra d'écrire l'opération en PYTHON `ip3 = ip1 & ip2`.

Ensuite tu pourras écrire une classe `Network`, initialisée avec une IP d'une machine sur le réseau (ou du réseau) et le nombre de bits du masque. Il sera alors possible d'obtenir le masque de sous-réseau, l'adresse de *broadcast*, tu peux même implémenter `__contains__` pour vérifier si une IP est une IP du réseau ou non...

Voici un exemple d'utilisation

```
ip1 = IP(192, 168, 181, 3)      # IP d'une machine sur un réseau
print(ip1.to_bin())             # affiche l'IP en binaire
```

```
n = Network(ip1, 20)             # réseau avec la machine, 20 bits fixes

print(n.mask_ip.to_bin())        # affiche le masque en binaire
print(n.mask_ip)                 # puis en décimal
print(n.network_ip.to_bin())     # IP du réseau en binaire
print(n.network_ip)              # puis en décimal
print(n.broadcast_ip.to_bin())   # IP de broadcast en binaire
print(n.broadcast_ip)            # puis en décimal

ip2 = IP(192,168,192,0)          # une deuxième IP
ip3 = IP(192,168,180,255)        # une troisième
print(ip2 in n)                  # ip2 est-elle sur le réseau ?
print(ip3 in n)                  # ip3 est-elle sur le réseau ?
```

Et voici le résultat

```
11111111.11111111.11110000.00000000
255.255.240.0
11000000.10101000.10110000.00000000
192.168.176.0
11000000.10101000.10111111.11111111
192.168.191.255
False
True
```