TD 2

A. Manipulation d'adresses

Le serveur "unserveur.org" a l'adresse IP 10.20.30.40

1. Quelle classe Java peut être utilisée pour représenter l'adresse du serveur ?

```
Class InetAddress
```

2. Écrire une méthode qui renvoie une instance de cette classe — à partir du nom de domaine, et — à partir de l'adresse IP.

```
import java.net.InetAddress;

InetAddress ia1 = InetAddress.getByName("unserveur.org");

InetAddress ia2 = InetAddress.getByAddress("10,20,30,40".getBytes());
```

B. Sockets UDP

1. Quelles sont les classes importantes pour UDP ?
    - `DatagramSocket`
    - `DatagramPacket`
2. Quelles informations sont-elles nécessaires à un client pour créer un DatagramPacket à destination d'un serveur ?

- le nombre d'octets à envoyer
- les données à envoyer
- l'adresse du serveur ou son nom
- le port du serveur

3. Écrire une méthode qui ouvre une socket UDP sur le port 5555, attend une connexion, lit le contenu du paquet reçu, l'affiche et ferme la socket.

```
DatagramSocket socket = new DatagramSocket(5555);

byte[] buffer = new byte[8192];

     DatagramPacket packet = new DatagramPacket(buffer, buffer.length);

     socket.receive(packet);

     String lu = new String(packet.getData());

     System.out.println(lu);

     socket.close();
```

C. Sockets TCP

1. Quelles sont les différences entre une transmission UDP et une transmission TCP ?

- UDP : "sans connexion", par paquet indépendant des autres, non fiable.
- TCP : "avec connexion", contrôle de flux, contrôle d'erreur, ordre conservé.

2. Quel avantage peut-t-il y avoir à utiliser UDP par rapport à TCP ?

- performances, moins de metadonnées pour le contrôle de flux, le contrôle d'erreurs, etc. Ex: application temps réél comme le transport de la voix

3. Détaillez, en 4 phases, le mécanisme Java de communication, coté client, entre un client et un serveur utilisant des sockets TCP.

   (1) Créér un socket (2) Demander la connexion au serveur (3) Recevoir/Envoyer des données (4) Fermer la socket

4. Écrire une méthode qui ouvre une socket TCP, la connecte au serveur myserveur.org sur le port 5555, lit la première ligne de données (supposées au format texte) envoyée par le serveur, l'affiche et ferme la socket.

```
Socket socket = new Socket(InetAddress.getByName("myserveur.org"),5555);

BufferedReader reader = new BufferedReader(new InputStreamReader(socket.getInputStream()));

String line = reader.readLine();

if(line != null) System.out.println(line);

socket.close();
```