PR6 – Programmation réseaux
TP n° 3 : Protocoles : découverte

Dans ce TP, toutes les transmissions seront faites en mode connecté TCP.

En Java, il est possible d'ouvrir une connexion à un service de *port* donné sur une *machine* donnée en utilisant la classe `java.net.Socket`. Par exemple :

```
Socket s = new Socket(machine,port);
```

Les méthodes `getOutputStream()` et `getInputStream()` permettent de retrouver des flux d'entrées/sorties (respectivement de type `OutputStream` et `InputStream`) pour écrire et lire sur cet objet de communication. De façon à manipuler ces objets, regardez comment les utiliser avec les classes `PrintStream`, `InputStreamReader` et `BufferedReader`.

Exercice 1 : Ports ouverts

Écrire un programme `Ports` en java qui teste quels ports, dont les numéros sont dans une plage donnée, sont ouverts sur une machine donnée. Par exemple la commande : `$ports 0-1024 lucien`
affiche sur la sortie standard tous les numéros de ports entre 0 et 1024 ouverts sur lucien et sur la sortie erreur ceux qui ne sont pas ouverts.

Exercice 2 : Client daytime

Écrire un programme `Jdaytime` en java qui se connecte au service `daytime` d'une machine donnée en paramètre et récupère la date pour l'afficher à l'écran.

Exercice 3 : Client/Serveur

1. Écrire un client et un serveur en Java. Le serveur choisira un entier aléatoirement entre 1 et 100. Et répondra « + » si son entier est plus grand que celui envoyé par le client, « - » s'il est plus petit et « = » s'il est égal avant de fermer la connexion. Le client affichera ce qu'il reçoit du serveur.
2. Sans changer le serveur, modifier votre client pour que ce soit plus lisible pour l'utilisateur.
3. Tester votre client avec le serveur d'un de vos camarades, et inversement.

Indications : La classe `InetAddress` vous permet d'avoir des informations sur la machine où vous exécutez votre programme, par exemple :
`InetAddress.getLocalHost().getHostName();`.
Pour créer le serveur utilisez la classe `ServerSocket`.

Exercice 4 : Serveur de mise en majuscules

Écrire un programme, `ServeurMajuscule`, représentant un serveur TCP itératif renvoyant les chaînes de caractères envoyées par des clients après les avoir mis en majuscule.
Le principe du serveur pour le traitement des requêtes d'une socket de service qu'il vient d'accepter, est le suivant :

- Envoyer au client un message d'accueil, précisant les modalités d'envoi des requêtes. Par exemple, les chaînes doivent être envoyées ligne par ligne, et l'envoi d'une ligne contenant uniquement un point (".") termine la session.
- Pour chaque ligne reçue par le client, le serveur la met en majuscule et la renvoie au client.
- Lorsqu'une ligne ne contenant qu'un point est reçue, le serveur renvoie cette ligne puis ferme la socket de service : il termine ainsi sa session avec ce client.

À son lancement, le serveur affiche son adresse et son numéro de port d'attachement local, afin que les clients puissent accéder à son service.