TD Sockets

Exercice 1 : extrait de l'examen 2011-2012 :

- Gros volume de données à charger depuis différents serveurs
- Pour gagner du temps : chargement de fragments différents depuis des serveurs différents (technique utilisée sur les plateformes de téléchargement P2P)
    - 3 serveurs
    - Client envoie le numéro du fragment demandé
    - Le serveur renvoie juste un message (fragment + numéro + numéro serveur)
    - Le client rassemble les messages reçus et les affiche

Ecrire en java les 2 programmes :

- USi.java avec numéro du serveur et numéro du port en arguments
- UC.java : charge 1 fragment depuis chacun des serveurs

Exercice 2 : Implanter un répartiteur de charge (*Load Balancer*) qui reçoit des requêtes HTTP et les redirige de façon aléatoire vers un ensemble de serveurs Web.

A chaque réception d'une requête HTTP (une connexion TCP), *LoadBalancer* transfère la requête à un des serveurs web (les adresses de ces serveurs sont données par les tables *hosts* et *ports*) et *LoadBalancer* transfère le résultat de la requête à l'émetteur.

Le choix du serveur Web est aléatoire (*rand.nextInt (nbHosts)* retourne un entier entre 0 et *nbHosts-1*).

Pour être efficace, *LoadBalancer* est multi-threadé.

Les échanges se feront en mode byte.

Le début de cette classe est le suivant :

```
public class LoadBalancer {

static String hosts[] = {"localhost", "localhost"};
static int ports[] = {8081,8082};
static int nbHosts = 2;
static Random rand = new Random();
…
}
```