UPMC
SORBONNE UNIVERSITÉS

SRCS : Systèmes Répartis Client/Serveur

TP 01 – Clavardage : des flux et du réseau.

Janvier 2016

À travers l'exemple d'un service de *clavardage* les étudiants seront progressivement amenés à utiliser les différents types de flux, à implémenter leur propre filtre et à utiliser des sockets TCP pour réaliser une première application répartie.

Exercice 1 : Mes premiers flux

Question 1

Écrire un programme java `Monologue` qui après avoir lu une phrase au clavier l'affiche à l'écran. Votre programme devra s'arrêter sur un `Ctrl+D`.

Question 2

On veut maintenant un programme `MonoLog` qui enregistre dans un fichier ce qui est saisi au clavier. Modifier votre premier programme pour enregistrer dans un fichier ce qui est lu sur le clavier tout en continuant à l'afficher à l'écran.

Question 3

Pour plus de "sécurité" on veut crypter ce qui est stocké sur le disque. Créer un filtre `CesarWriter` qui décale circulairement les caractères d'un entier passé en argument au constructeur. Par exemple pour un décalage de 5 :
`'a'` → `'f'`,`'k'` → `'p'`,`'x'` → `'c'`.

Cette classe héritera de `FilterWriter` en redéfinissant la méthode `write(String,int,int)`.

Question 4

Utiliser votre filtre `CesarWriter` dans un programme `CryptoLog` qui chiffrera les données lues sur votre clavier avant de les écrire dans votre fichier de log.

Question 5

À partir des classes déjà réalisées, implémenter en deux lignes un programme `Chiffrement` qui génère une version chiffrée (par l'algorithme de Cesar) d'un fichier passé en paramètre. Le nom de ce nouveau fichier sera suffixé par `'.Cesar'`.

Exercice 2 : Clavardage sur TCP

Dans cet exercice on veut implémenter un service de *clavardage* entre deux ordinateurs distants reliés par une connexion TCP.

Question 1

Coder un client et un serveur qui établissent une connection TCP et affiche sur leur terminal respectif leur port de dialogue et celui de leur interlocuteur.

Question 2

A l'aide d'un temporisateur utilisant la méthode de classe `Thread.sleep`, traiter le cas ou le client tente de se connecter avant que le serveur ne soit réceptif. Votre code devra éviter que le client ne s'acharne sur le réseaux. Il devra donc attendre un certain temps entre chaque tentative. Vous veillerez aussi à ne pas confondre une erreur d'adresse avec un echec de connexion.

Question 3

Maintenant modifier votre serveur pour qu'il accepte successivement la connexion de trois clients avant de se terminer.

Question 4

Implémenter une première version du clavardage où seul le client envoie des messages. Ces messages sont lus sur le clavier du client, envoyés au serveur qui les affichera sur son terminal.

Question 5

Utiliser votre filtre `CesarWriter` pour crypter la communication entre le client et le serveur. Votre programme pourra afficher sur le terminal du serveur le message chiffré puis sa version déchiffrée.

Question 6

Que se passe-t-il si le serveur reprend le code du client pour envoyer des messages ?

Question 7

On veut permettre aux deux participants de communiquer. Proposer un protocole qui, *sous certaines conditions*, assure que la communication puisse se faire dans les deux sens, *i.e.* que des messages du client (resp. du serveur) puissent s'afficher sur le terminal du serveur (resp. du client).

