NET4102

TP4

Le but de ce TP est de se familiariser avec des fonctions de mulitplexage de connexions select() et poll().

Exercice 1

Ecrivez un programme client/serveur en TCP permettant de faire l'echo des messages sur le port 8888. Le client reste connecté tant que l'utilisateur n'a pas tapé le message « quit# ».

Votre serveur doit comptabiliser le nombre de caractères reçus, appliquer une fonction de cout, et renvoyer par la suite le résultat sous la forme d'une structure composée de trois champs, qui reprend le message, rajoute le nombre de caractères, et le cout total correspondant. Comme fonction de cout vous pouvez considérer la fonction suivante :

Cout (exprimé en Eur)= nombre de caractères x 0,23 (Eur).

Lancez le serveur, ouvrez deux terminaux, exécutez le programme client dans chacune des fenêtres, et essayez d'envoyer des messages vers le serveur.

Exercice 2

Dans cet exercice on vous demande de modifier votre programme afin de gérer plusieurs connexions en parallèle en utilisant la fonction select(). Par ailleurs, nous souhaitons que le serveur affiche l'heure par tranche de 30 secondes. De plus on souhaite pouvoir modifier le cout en cours d'exécution du serveur sans le relancer. Il suffit de taper en ligne de commande un message sous la forme suivante « *C*XXX » avec XXX le cout exprimé en centimes d'Eur.

Exercice 3

Dans cet exercice, on vous demande de modifier votre programme et d'utiliser la fonction poll() à la place de la fonction select().

Pseudo-algorithme incomplet utilisant la fonction select()

Début Algorithme

Déclaration et initialisation des variables, des ensembles des descripteurs et des sockets nécessaires

Boucle infinie
{
FD1←FD1master // initialiser l'ensemble FD1 qui sera altéré après l'appel de la fonction
// select()

select(MaxsockDesc+1, FD1,FD2,FD3,timeoutset) // FD2 et FD3 seront mis a NULL pour
// notre exercice
si (timeoutset à expiré)
{
Traitement timeoutset
Réarmer le timeoutset
}
sinon
{
Parcourir l'ensemble des descripteurs FD1

- Pour chaque descripteur armé
  si (évènement provenant du descripteur de la socket en écoute)
  {
  Accepter la nouvelle connexion (avec la fonction accept())
  Ajouter nouveau descripteur à l'ensemble des descripteurs à écouter
  Ajuster la variable MaxsockDesc si nécessaire
  }
  Sinon
  {
  Des données sont prêtes à être lues sur le descripteur en question
  Exécuter le traitement nécessaire, et répondre éventuellement au client
  }

}
}

Fin Algorithme