TD 3 – Programmation Socket en C

1 – Familiarisation avec la programmation

```
#include <stdio.h>
#include <stdlib.h>
#include <errno.h>
#include <string.h>
#include <netinet/in.h>
#include <sys/socket.h>
#include <sys/types.h>

#define PORT 12345
int sock, socket2, lg;
char mess[80];
struct sockaddr_in local;   // champs d'entete local
struct sockaddr_in distant; // champs d'entete distant

creer_socket()
{
// preparation des champs d'entete
   bzero(&local, sizeof(local));        // mise a zero de la zone adresse
   local.sin_family = AF_INET;          // famille d adresse internet
   local.sin_port = htons(PORT);        // numero de port
   local.sin_addr.s_addr = INADDR_ANY;  // types d'adresses prises en charge
   bzero(&(local.sin_zero),8);          // fin de remplissage

   lg = sizeof(struct sockaddr_in);
// creation socket du serveur mode TCP/IP
   if((sock=socket(AF_INET, SOCK_STREAM,0)) == -1){perror("socket"); exit(1);}
}

main()
{
   // creation socket
   creer_socket();
   // nommage de la socket
   if(bind(sock, (struct sockaddr *)&local, sizeof(struct sockaddr)) == -1)
     {perror("bind");exit(1);}
   // mise a l'ecoute
   if(listen(sock, 5) == -1){perror("listen");exit(1);}

   // boucle sans fin pour la gestion des connexions
   while(1)
     {  // attente connexion client
        if((socket2=accept(sock, (struct sockaddr *)&distant, &lg)) == -1)
        {perror("accept");exit(1);}      // En cas de pb on quitte le programme
        printf ("client connecte \n");
         read(socket2,mess,80);           // Lecture flux de données
         printf ("le client me dit %s \n",mess);
        close(socket2);                  // Fermeture/libération de la socket
     }
}
```

Découverte
Commentez les parties notées ???

Test en local:
Saisir, compilez et exécutez le programme ci-dessus.
Ouvrez une autre console et lancez : **telnet 127.0.0.1 12345.** Que se passe-t-il ?

2 – Familiarisation avec la programmation

Complétez le programme ci-dessous pour qu'il puisse communiquer avec le programme précédent.

```
#include <stdio.h>
#include <netdb.h>
#include <fcntl.h>
#include <string.h>
#include <sys/socket.h>
#include <netinet/in.h> // gestion adresses i
#include <sys/types.h>

#define SERV 127.0.0.1    // adresse IP = boucle locale
#define PORT 12345        // port d'ecoute serveur
int port,sock;            // n°port et socket
char mess[80];            // chaine de caracteres

struct  sockaddr_in     serv_addr;      // zone adresse
struct  hostent         *server;        // nom serveur

creer_socket()
{  port = PORT;
  server = gethostbyname(SERV);         // verification existance adresse
  if (!server){fprintf(stderr, "Problème serveur \"%s\"\n", SERV);exit(1);}
  // creation socket locale
  sock = socket(AF_INET, SOCK_STREAM, 0);  // creation socket
  bzero(&serv_addr, sizeof(serv_addr));   // preparation champs entete
  serv_addr.sin_family = AF_INET;         // Type d'adresses
  bcopy(server->h_addr, &serv_addr.sin_addr.s_addr,server->h_length);
  serv_addr.sin_port = htons(port);       // port de connexion du serveur
}

main()
{ // creation socket
  creer_socket();
  // connexion au serveur
  // connexion à l'application du dessus
  if (connect(sock, (struct sockaddr *)&serv_addr, sizeof(serv_addr)) < 0)
   {perror("Connexion impossible:");exit(1);}
  printf ("connexion avec serveur ok\n");

  // saisie d'une chaine de caractères au clavier
  printf ("Entrer un message : ");
  gets(mess);

  // envoi de cette chaine à l'application du dessus
  write(sock,mess,80);

  close (sock);
}
```

Si tout fonctionne, BRAVO, vous venez de créer votre première application Client/Serveur !!!
Mais au fait qui est client, qui est le serveur ? Justifiez votre réponse .
Le serveur est l'application qui nomme la socket (bind) et qui attends la connexion (accept), donc la première de ce TD.