

TP3 Client/serveur TCP

M1 IRCOMS

Exercice 1 :

- 1- Compléter le code suivant pour réaliser une communication client serveur avec le protocole TCP (mode connecté).

client.c

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

int main()
{
    int sock, bytes_recieved;
    char send_data[1024],rcv_data[1024];
    struct hostent *host;
    struct sockaddr_in server_addr;

    host = gethostbyname("127.0.0.1");

    if ((sock = ... == -1) {
        perror("Socket");
        exit(1);
    }

    server_addr.sin_family = ...
    server_addr.sin_port = ...
    server_addr.sin_addr = ...
    bzero(&(server_addr.sin_zero),8);

    if (... == -1)
    {
        perror("Connect");
        exit(1);
    }

    while(1)
    {
        bytes_recieved=...;
        rcv_data[bytes_recieved] = '\0';

        if (strcmp(rcv_data , "q") == 0 || strcmp(rcv_data , "Q") == 0)
        {
            close(sock);
            break;
        }
    }
}
```

```
else
    printf("\nRecieved data = %s ", recv_data);

    printf("\nSEND (q or Q to quit) : ");
    gets(send_data);

    if (strcmp(send_data , "q") != 0 && strcmp(send_data , "Q") != 0)
        ...;

    else
    {
        ....;
        close(sock);
        break;
    }

}
return 0;
}
```

server.c

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

int main()
{
    int sock, connected, bytes_recieved , true = 1;
    char send_data [1024] , recv_data[1024];

    struct sockaddr_in server_addr,client_addr;
    int sin_size;

    if ((sock = ... == -1) {
        perror("Socket");
        exit(1);
    }

    server_addr.sin_family = ...;
    server_addr.sin_port ...;
    server_addr.sin_addr.s_addr = INADDR_ANY;
    bzero(&(server_addr.sin_zero),8);

    if (... == -1) {
        perror("Unable to bind");
        exit(1);
    }

    if (... == -1) {
```

```
    perror("Listen");
    exit(1);
}

    printf("\nTCPServer Waiting for client on port 5000");
    fflush(stdout);

while(1)
{

    sin_size = sizeof(struct sockaddr_in);

    connected = accept(sock, (struct sockaddr *)&client_addr,&sin_size);

    printf("\n I got a connection from (%s , %d)",
        inet_ntoa(client_addr.sin_addr),ntohs(client_addr.sin_port));

    while (1)
    {
        printf("\n SEND (q or Q to quit) : ");
        gets(send_data);

        if (strcmp(send_data , "q") == 0 || strcmp(send_data , "Q") == 0)
        {
            send(connected, send_data,strlen(send_data), 0);
            close(connected);
            break;
        }

        else
            send(connected, send_data,strlen(send_data), 0);

        bytes_recieved = ...

        recv_data[bytes_recieved] = '\0';

        if (strcmp(recv_data , "q") == 0 || strcmp(recv_data , "Q") == 0)
        {
            close(connected);
            break;
        }

        else
            printf("\n RECIEVED DATA = %s " , recv_data);
            fflush(stdout);
        }
    }

    close(sock);
    return 0;
}
```