**ESAME 21/09/2012 – SERVER CONCORRENTE**

#include <arpa/inet.h>

#include <fcntl.h>

#include <netdb.h>

#include <netinet/in.h>

#include <signal.h>

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <sys/ioctl.h>

#include <sys/socket.h>

#include <sys/types.h>

#include <unistd.h>

#include <sys/select.h>

#define PORT 2109 //Numero sceglilo tu

void handler(int signo) //Se hai 2 segnali passi signo altrimenti no.

{}

**int main**(int argc, char \*argv[]) //VARIABILI OBBLIGATORIE

{

int sock, msgsock, lenght, piped[2]\*; // QUESTI SOLO PER SOCKET, piped solo se servono le pipe: generalmente la dimensione è 2, altrimenti matrice di pipe.

struct sockaddr\_in server, client; // Servono per i server;

struct sigaction act; //inizializzo gestore segnale sigusr

int on = 1; //Serve per il socket (serve per il setsockopt)

q = atoi(argv[1]); // ottengo il parametro q da input (per interi ricordarsi di usare atoi)

if (pipe(piped) < 0) { //creazione di una pipe con debug

perror("Creazione pipe");

exit(3);

}

**Gestione affidabile dei segnali**

sigemptyset(&act.sa\_mask); // azzero la mask

act.sa\_handler = handler;

act.sa\_flags = SA\_RESTART;

if ((sigaction(SIGUSR1, &act, NULL)) <

0) // applico la sigaction act al SIGUSR1

{

perror("Errore sigaction SIGUSR1");

exit(4);

}

if ((sigaction(SIGUSR2, &act, NULL)) <

0) // applico la sigaction act al SIGUSR2

{

perror("Errore sigaction SIGUSR1");

exit(-1);

}

**SOCKET**

if ((sock = socket(AF\_INET, SOCK\_STREAM, 0)) < 0) // creazione socket

{

perror("Creazione stream socket");

exit(5);

}

int n;

do {

n= select(sock + 1, &sock\_fdset, NULL, NULL, NULL);

}tet

while(n==-1 && errno== EINTR); //test di lettura sulla socket

}

**CONFIGURAZIONE SERVER**

server.sin\_family = AF\_INET;

server.sin\_addr.s\_addr = INADDR\_ANY; // Imposto i parametri della socket

server.sin\_port = htons(PORT);

if (setsockopt(sock, SOL\_SOCKET, SO\_REUSEADDR, &on, sizeof(on)) < 0) {

perror("setsockopt");

}

if (bind(sock, (struct sockaddr \*)&server, sizeof(server)) < 0) {

perror("Binding stream socket");

exit(6);

}

lenght = sizeof(server);

if (getsockname(sock, (struct sockaddr \*)&server, (socklen\_t \*)&lenght) <0) // leggo l'indirizzo effettivo della socket

{

perror("Getting socket name");

exit(7);

}

printf("Porta della socket # %d\n", ntohs(server.sin\_port));

listen(sock, 4);

printf("In attesa di connessioni...\n");

do // attende connessioni e crea un figlio per ogni nuovo cliente che si connette  
{  
msgsock = accept(sock, (struct sockaddr \*)&client,  
(socklen\_t \*)&lenght); // attendo connessioni...

if (msgsock == -1) {  
perror("accept");  
exit(6);

}

do {

// Questa ioctl ritorna in available

// il numero di byte disponibili

// in lettura dalla pipe

ioctl(piped[0], FIONREAD, &available);

if (available > 0)

read(piped[0], &m, sizeof m);

} while(available >0);

// quando utilizzi fork dentro l’if UTILIZZARE close(sock).

PER SCRIVERE SUL SOCKET MENTRE SEI DENTRO UN PROCESSO FIGLIO BISOGNA CREARE UN MSGSOCK E SCRIVERCI DENTRO.

}

while(1);