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

#define MAXLINE 4096
#define max(a,b) \
  ({ __typeof__ (a) _a = (a); \
    __typeof__ (b) _b = (b); \
    _a > _b ? _a : _b; })

void sendCommand(int sockfd, const char* cmd) {
  write(sockfd, cmd, strlen(cmd));
}

void echoServerAnswer(int sockfd) {
  ssize_t n;
  char recvline[MAXLINE + 1] = "";
  //leia MAXLINE bytes do socket
  n = Read(sockfd, recvline, MAXLINE);
  //escreva na tela
  if (n ≡ 0) return;
  Fputs(recvline, stdout);
}

int main(int argc, char **argv) {
   int    sockfd;
   char   error[MAXLINE + 1];
   struct sockaddr_in servaddr;

   //trate os argumentos
   if (argc ≠ 3) {
      //usage
      strcpy(error,"uso: ");
      strcat(error,argv[0]);
      strcat(error," <IPaddress> <Porta>");
      perror(error);
      exit(EXIT_FAILURE);
   }

   //crie um socket para comunicacao, e aborte em caso de erro.
   sockfd = Socket(AF_INET, SOCK_STREAM, 0);
   //parametros de socket
   bzero(&servaddr, sizeof(servaddr)); //inicialize com zeros
   servaddr.sin_family = AF_INET; //servidor de enderecos IPv4
   servaddr.sin_port   = htons(atoi(argv[2])); //Porta como argumento
   //converta o endereco IP de texto para binario. Reporte erros
   Inet_pton(AF_INET, argv[1], &servaddr.sin_addr);

   //conecte o socket com o endereco passado por argumento
   Connect(sockfd, (struct sockaddr *) &servaddr, sizeof(servaddr));

   struct sockaddr_in getsock;
   socklen_t addrlen = sizeof(struct sockaddr);
   //obtenha o endereco com o qual estamos comunicando
   Getsockname(sockfd, (struct sockaddr*) &getsock, &addrlen);
   //imprima o endereco no stdout
   printf("Connected to server: %s:%d\n",
     inet_ntoa(getsock.sin_addr), ntohs(getsock.sin_port));

  char recvline[MAXLINE + 1] = "";
```



```c
  char sendline[MAXLINE + 1] = "";
  fd_set rset;
  FD_ZERO(&rset);
  while(1) {
    FD_SET(fileno(stdin), &rset);
    FD_SET(sockfd, &rset);
    int maxfdp1 = max(fileno(stdin), sockfd) + 1;
    Select(maxfdp1, &rset, NULL, NULL, NULL);
    if (FD_ISSET(sockfd, &rset)) { //atividade no socket
      int n = Read(sockfd, recvline, MAXLINE);
      if (n ≡ 0) {
        fprintf(stderr, "server terminated prematurely");
        exit(EXIT_FAILURE);
      }
      Fputs(recvline, stdout);
    }
    if (FD_ISSET(fileno(stdin), &rset)) { //atividade na entrada padrao
      if (fgets(sendline, MAXLINE, stdin) ≡ NULL) {
      } else sendCommand(sockfd, sendline);
    }
    fflush(stdout);
  }
  return 0;
}
```