```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 "errorHandling.c"

#define MAXLINE 4096

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;
  printf("server answer:\n");
  Fputs(recvline, stdout);
}

char exitCommand(const char* line) {
  char ret = (strcmp(line,"exit") ≡ 0) ∨
    (strcmp(line, "bye")      ≡ 0) ∨
    (strcmp(line, "sair")     ≡ 0) ∨
    (strcmp(line, "quit")     ≡ 0);
  return ret;
}

void removeEnter(char *line) {
  if (line[strlen(line) − 1] ≡ '\n') line[strlen(line) − 1] = '\0';
}

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
```



```c
   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));

  ssize_t r;
  do {
    char *line = NULL;
    size_t len = 0;

    r = getline(&line, &len, stdin);
    removeEnter(line);
    if (r > 0) {
      //printf("local %zu bytes input:%s", r, line);
      printf("local input:%s\n", line);
      if (exitCommand(line)) {
        printf("Encerrando conexao com o servidor...\n");
        r = -1;
      } else if (strcmp(line, "\n") ≠ 0) {
        sendCommand(sockfd, line );
        echoServerAnswer(sockfd);
      }
    }
  } while(r ≠ -1);
  close(sockfd);

   return 0;
}
```