```c
// uxdstrsock.h

#define SOCKNAME "/tmp/ourdemosocket"

//UNIX Domain Stream Socket Server (Example)

#include <sys/un.h>
#include <sys/socket.h>
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <string.h>
#include <ctype.h>
#include "uxdstrsock.h"

#define BUFSIZE 1024*1024

void err_exit (char *msg){
  perror(msg);
  exit(EXIT_FAILURE);
}

void transmogrify(char *buf, int buflen){
  int c, x;

  for (c=0; c<buflen; c++) {
    x = buf[c];
    if (isalpha(x)) {
      if ((x<'X') || ((x>='a')&&(x<'x')))
        buf[c]+=3;
      else
        buf[c]-=23;
    }
  }
}

int main(int argc, char *argv[]){
    struct sockaddr_un addr;
    int sfd, cfd, l, l2, c;
    char buf[BUFSIZE];

    if ((sfd = socket(AF_UNIX, SOCK_STREAM, 0)) == -1) {
      err_exit("socket()");
    }

    memset(&addr, 0, sizeof(struct sockaddr_un));
    addr.sun_family = AF_UNIX;
    strcpy(addr.sun_path, SOCKNAME);
    if (bind(sfd, (struct sockaddr*) &addr, sizeof(addr)) == -1) {
      err_exit("bind()");
    }
    if (listen(sfd, 2) == -1) {
        err_exit("listen()");
    }

    while (1) { /* Iterative Server Loop */
      if ((cfd = accept(sfd, NULL, NULL)) == -1) {
            err_exit("accept()");
      }

      l = read(cfd, buf, BUFSIZE);
      while (l > 0) {
        transmogrify(buf, l);
        l2 = write(cfd, buf, l);
        if (l != l2) {
          printf("partial/failed write");
          break;
        }
        l = read(cfd, buf, BUFSIZE);
      }
      if (l == -1) {
        err_exit("read()");
      }
      close(cfd);
    }
    unlink(SOCKNAME);
    exit(EXIT_SUCCESS);
}

//UNIX Domain Stream Socket Client (Example)

#include <sys/un.h>
#include <sys/socket.h>
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <string.h>
#include "uxdstrsock.h"

#define BUFSIZE (1024*1024)

void err_exit (char *msg){
  perror(msg);
  exit(EXIT_FAILURE);
}

int main(int argc, char *argv[]){
    struct sockaddr_un addr;
    int sfd, ret, l, l2;
    char buf[BUFSIZE];

    if ((sfd = socket(AF_UNIX, SOCK_STREAM, 0)) == -1) {
      err_exit("socket()");
    }

    memset(&addr, 0, sizeof(struct sockaddr_un));
    addr.sun_family = AF_UNIX;
    strcpy(addr.sun_path, SOCKNAME);

    if ((ret = connect(sfd, (struct sockaddr *) &addr, sizeof(addr))) == -1) {
        err_exit("connect()");
    }

    /* Copy stdin to socket */

    l = read(STDIN_FILENO, buf, BUFSIZE);
    while (l > 0) {
      l2 = write(sfd, buf, l);
      if (l2 != l) {
            printf("partial/failed write()");
      }
      read(sfd, buf, l2);  /* XXX Error Handling, dammit! */
      printf("%s", buf);
      l = read(STDIN_FILENO, buf, BUFSIZE);
    }
    if (l == -1) {
        err_exit("read(), stdin");
    }
    close(sfd);
    exit(EXIT_SUCCESS);
}
```