Studio5

**Exercise1:** Qitao Xu, Jiangnan Liu, Zhe Wang

**Exercise2**: Code for server

#include <stdio.h> //For printf()

#include <stdlib.h> //for exit() and atoi()

#include <signal.h>

#include <string.h>

#include <errno.h>

#include <unistd.h> // for write()

#include <sys/types.h>

#include <sys/socket.h> // for bind(), socket()

#include <sys/un.h> // for unix()

#define SOCKET\_PATH "/home/pi/Socket"

#define LISTEN\_BACKLOG 50

#define BUF\_SIZE 50

char buf[BUF\_SIZE];

int main( int argc, char\* argv[] ){

int skt, ret\_bind, ret\_listen, accept\_skt, ret\_unlink, ret\_read;

struct sockaddr\_un skt\_addr, peer\_addr;

socklen\_t peer\_addr\_size;

skt = socket(AF\_UNIX, SOCK\_STREAM, 0);

if (skt < 0) {

printf("Error: socket() system call failed! Reason: %s\n", strerror(errno));

exit(-1);

}

memset(&skt\_addr, 0, sizeof(struct sockaddr\_un));

skt\_addr.sun\_family = AF\_UNIX;

strncpy(skt\_addr.sun\_path, SOCKET\_PATH, strlen(SOCKET\_PATH));

ret\_bind = bind(skt, (struct sockaddr \*)&skt\_addr, sizeof(struct sockaddr\_un));

if (ret\_bind < 0) {

printf("Error: bind() system call failed! Reason: %s\n", strerror(errno));

exit(-1);

}

ret\_listen = listen(skt, LISTEN\_BACKLOG);

if (ret\_listen < 0) {

printf("Error: listen() system call failed! Reason: %s\n", strerror(errno));

exit(-1);

}

peer\_addr\_size = sizeof(struct sockaddr\_un);

accept\_skt = accept(skt, (struct sockaddr \*)&peer\_addr, &peer\_addr\_size);

if (accept\_skt < 0) {

printf("Error: accept() system call failed! Reason: %s\n", strerror(errno));

exit(-1);

}

memset(buf, 0, BUF\_SIZE);

while(1) {

     ret\_read = read(accept\_skt, buf, BUF\_SIZE);

if (ret\_read > 0) {

printf("%s", buf);

memset(buf, 0, BUF\_SIZE);

}

}

ret\_unlink = unlink(skt\_addr.sun\_path);

if (ret\_unlink < 0) {

printf("Error: unlink() system call failed! Reason: %s\n", strerror(errno));

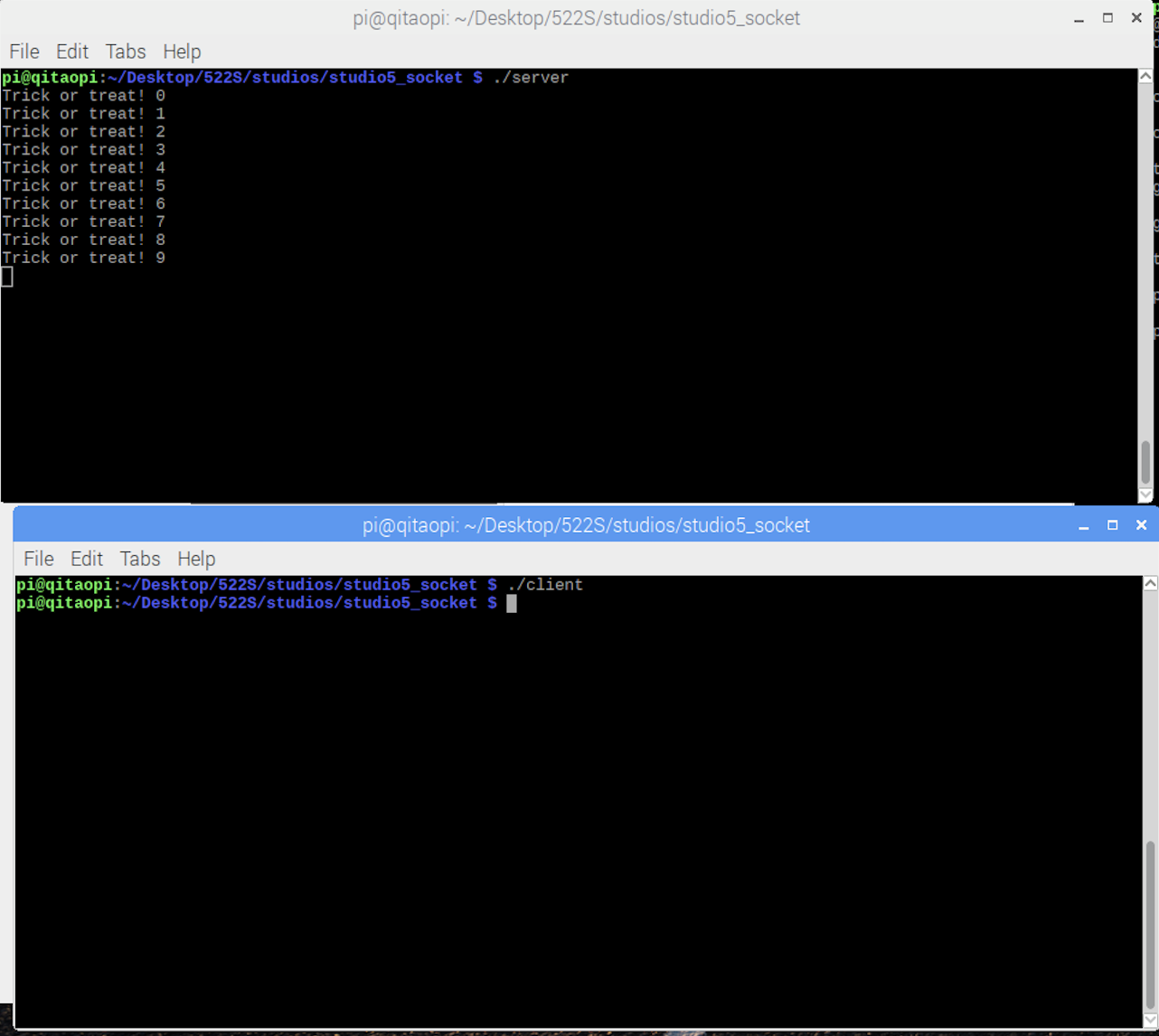
exit(-1);

}

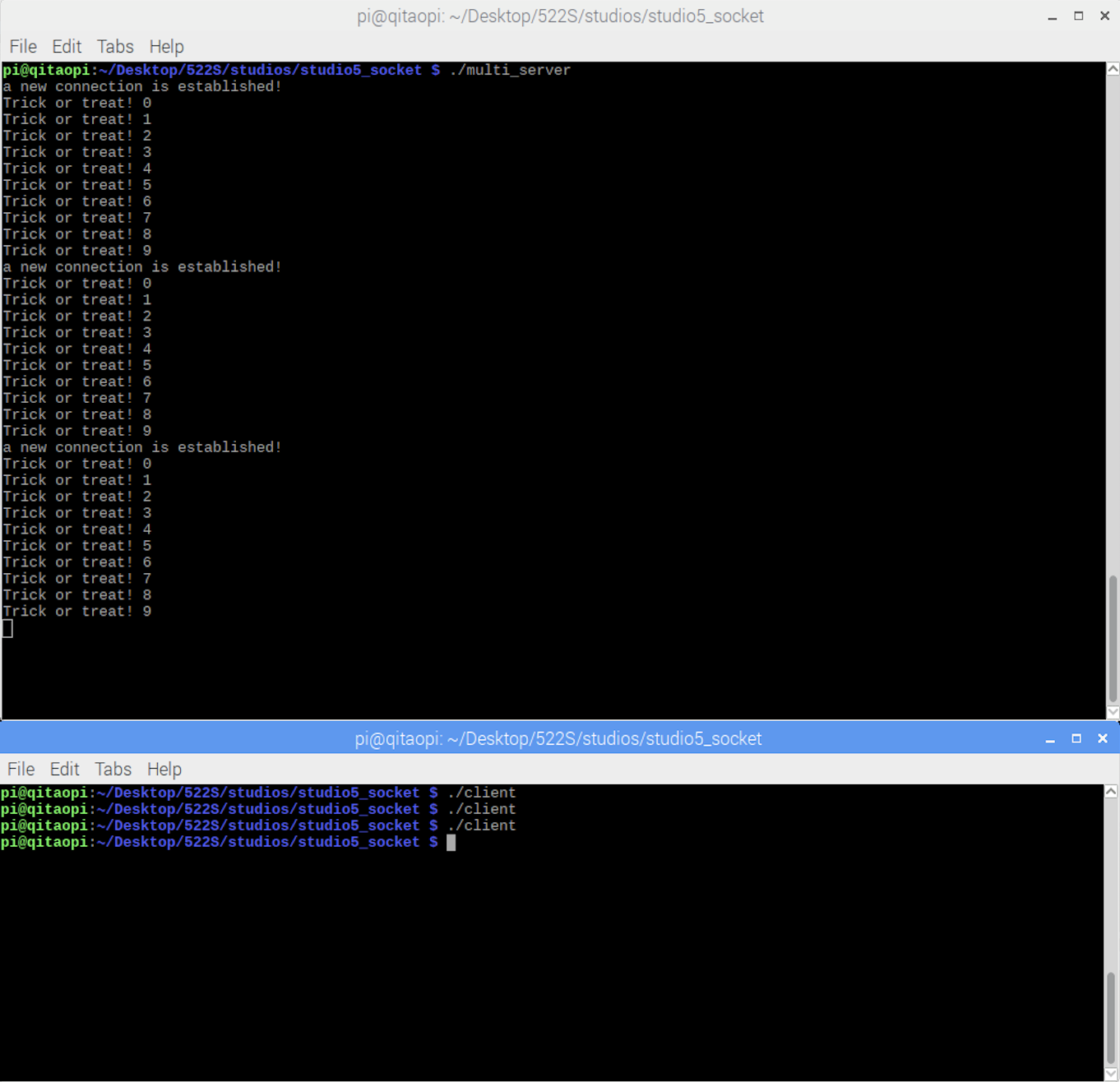
return 0;

}

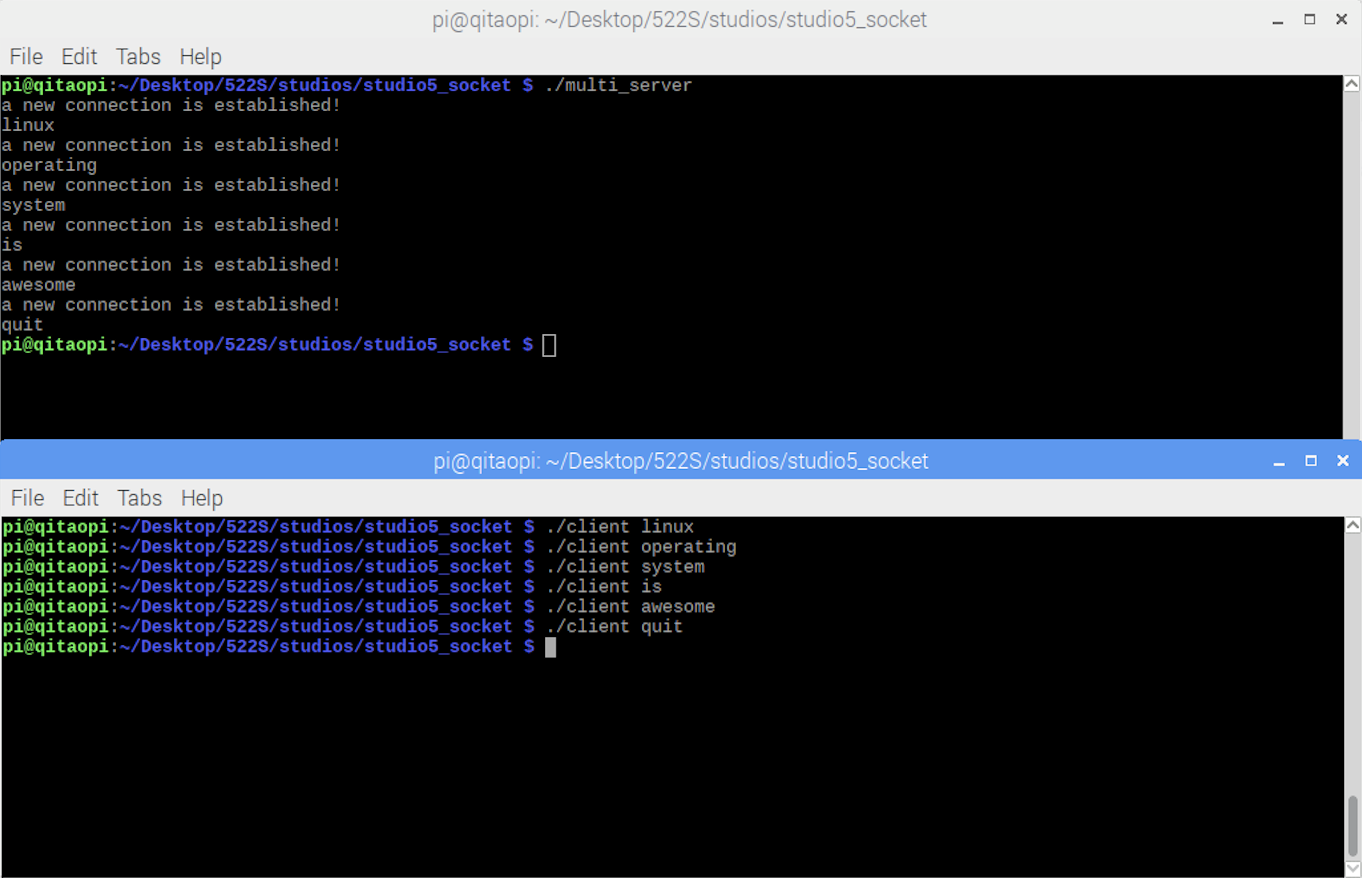
**Exercise3**:



**Exercise4**:



**Exercise5**:



**Exercise6**:



