//Design a TCP concurrent server to echo given set of sentences using poll

functions

//server program

#include <stdlib.h>

#include <stdio.h>

#include <string.h>

#include <sys/types.h>

#include <sys/stat.h>

#include <sys/socket.h>

#include <netinet/in.h>

#include <arpa/inet.h>

#include <fcntl.h>

#include <limits.h>

#include <poll.h>

#include <errno.h>

#define MAXLINE 100

#define SERV\_PORT 13154

#define POLLRDNORM 5

#define INFTIM 5

#define OPEN\_MAX 5

int main(int argc, char \*\*argv)

{

int k, i, maxi, listenfd, connfd, sockfd;

int nready;ssize\_t n;

char line[MAXLINE];

socklen\_t clilen;

struct pollfd client[OPEN\_MAX];

struct sockaddr\_in cliaddr, servaddr;

listenfd = socket(AF\_INET, SOCK\_STREAM, 0);

bzero(&servaddr, sizeof(servaddr));

servaddr.sin\_family = AF\_INET;

servaddr.sin\_addr.s\_addr = htonl(INADDR\_ANY);

servaddr.sin\_port = htons(SERV\_PORT);

bind(listenfd, (struct sockaddr \*) &servaddr, sizeof(servaddr));listen(listenfd, 5);

client[0].fd = listenfd;

client[0].events = POLLRDNORM;

for (i = 1; i < OPEN\_MAX; i++)

client[i].fd = -1;

maxi = 0;

for ( ; ; )

{

nready = poll(client, maxi+1, INFTIM);

if (client[0].revents & POLLRDNORM)

{

clilen = sizeof(cliaddr);

connfd = accept(listenfd, (struct sockaddr \*) &cliaddr, &clilen);

#ifdef NOTDEF

printf("new client: %s\n", sock\_ntop((struct sockaddr \*) &cliaddr,clilen));

#endif

for (i = 1; i < OPEN\_MAX; i++)

if (client[i].fd < 0)

{

client[i].fd = connfd;

break;

}if (i == OPEN\_MAX)

{

printf("too many clients");

exit(0);

}

client[i].events = POLLRDNORM;

if (i > maxi)maxi = i;

if (--nready <= 0)

continue;

}

for (i = 1; i <= maxi; i++)

{

if ( (sockfd = client[i].fd) < 0)

continue;

if (client[i].revents & (POLLRDNORM | POLLERR))

{

if ( (n = read(sockfd, line, MAXLINE)) < 0)

{

if (errno == ECONNRESET)

{

#ifdef NOTDEF

printf("client[%d] aborted connection\n", i);

#endif

close(sockfd);

client[i].fd = -1;

}

else

printf("readline error");

}

else if (n == 0)

{

#ifdef NOTDEF

printf("client[%d] closed connection\n", i);

#endif

close(sockfd);

client[i].fd = -1;

}

else{ printf("\n data from client is \n");

k=strlen(line);

printf(" length=%d data = %s\n", k,line);

strcpy(line," ");

}

if (--nready <= 0)

break;

}

}

}

}