**//Design a TCP concurrent server to convert a given text into upper case using multiplexingsystem call**

**//server program**

**#include <stdio.h>**

**#include <stdlib.h>**

**#include <unistd.h>**

**#include <errno.h>**

**#include <string.h>**

**#include <sys/types.h>**

**#include <sys/stat.h>**

**#include <sys/select.h>**

**#include <sys/time.h>**

**#include <sys/socket.h>**

**#include <netinet/in.h>**

**#include <arpa/inet.h>**

**#include <fcntl.h>**

**#define MAXLINE 100**

**#define SERV\_PORT 13153**

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

**{**

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

**int nready, client[FD\_SETSIZE];**

**ssize\_t n;**

**fd\_set rset, allset;**

**char line[MAXLINE],buf[100];**

**socklen\_t clilen;**

**struct sockaddr\_in cliaddr, servaddr;**

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

**if (listenfd < 0 )**

**{**

**perror("socket" );**

**exit(1);**

**}**

**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);**

**maxfd = listenfd; /\* initialize \*/**

**maxi = -1; /\* index into client[] array \*/**

**for (i = 0; i < FD\_SETSIZE; i++)**

**client[i] = -1; /\* -1 indicates available entry \*/**

**FD\_ZERO(&allset);**

**FD\_SET(listenfd, &allset);/\* end fig01 \*//\* include fig02 \*/**

**for ( ; ; )**

**{**

**printf("\n Server:I am waiting-----Start of Main Loop\n");**

**rset = allset; /\* structure assignment \*/**

**nready = select(maxfd+1, &rset, NULL, NULL, NULL);**

**if (FD\_ISSET(listenfd, &rset))**

**{ /\* new client connection \*/**

**clilen = sizeof(cliaddr);**

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

**#ifdef NOTDEF**

**printf("new client: %s, port %d\n",inet\_ntop(AF\_INET, &cliaddr.sin\_addr, buf, NULL),ntohs(cliaddr.sin\_port));**

**#endif**

**for (i = 0; i < FD\_SETSIZE; i++)**

**if (client[i] < 0)**

**{**

**client[i] = connfd; /\* save descriptor \*/**

**break;**

**}if (i == FD\_SETSIZE)**

**{**

**printf("too many clients");**

**exit(0);**

**}**

**FD\_SET(connfd, &allset); /\* add new descriptor to set \*/**

**if (connfd > maxfd)**

**maxfd = connfd; /\* for select \*/**

**if (i > maxi)**

**maxi = i; /\* max index in client[] array \*/**

**if (--nready <= 0)**

**continue; /\* no more readable descriptors\*/**

**}**

**for (i = 0; i <= maxi; i++)**

**{ /\* check all clients for data \*/**

**if ( (sockfd = client[i]) < 0)**

**continue;**

**if (FD\_ISSET(sockfd, &rset))**

**{**

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

**{/\*4connection closed by client \*/**

**close(sockfd);**

**FD\_CLR(sockfd, &allset);**

**client[i] = -1;**

**}**

**else**

**{**

**printf("\n output at server\n");**

**for(k=0;line[k]!='\0';k++)**

**printf("%c",toupper(line[k]));**

**write(sockfd, line, n);**

**}if (--nready <= 0)**

**break; /\* no more readable descriptors\*/**

**}}}}**