PROGRAM-07

//CLIENT

#include<sys/types.h>

#include<sys/socket.h>

#include<netinet/in.h>

#include<sys/stat.h>

#include<unistd.h>

#include<stdlib.h>

#include<stdio.h>

#include<fcntl.h>

#include<arpa/inet.h>

void str\_cli(FILE \*fp, int sockfd)

{

int bufsize=1024, cont;

char \*buffer=malloc(bufsize);

while(fgets(buffer,bufsize,fp)!=NULL)

{

send(sockfd, buffer, sizeof(buffer),0);

if((cont=recv(sockfd,buffer,bufsize,0))>0)

{

//fputs(buffer,stdout);

}

}

printf("\nEOF\n");

}

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

{

int create\_socket;

struct sockaddr\_in address;

if((create\_socket=socket(AF\_INET,SOCK\_STREAM,0))>0)

printf("The socket was created.\n");

address.sin\_family=AF\_INET;

address.sin\_port=htons(15001);

inet\_pton(AF\_INET, argv[1], &address.sin\_addr);

if(connect(create\_socket,(struct sockaddr\*)&address, sizeof(address))==0)

printf("The connection was accepted with the server %s...\n",argv[1]);

else

printf("Error in connect()\n");

str\_cli(stdin, create\_socket);

return close(create\_socket);

}

//SERVER

#include<sys/types.h>

#include<sys/socket.h>

#include<netinet/in.h>

#include<sys/stat.h>

#include<unistd.h>

#include<stdlib.h>

#include<stdio.h>

#include<fcntl.h>

#include<arpa/inet.h>

void remote\_command(int connfd, int port)

{

int n;

int bufsize=1024;

char \*buffer=malloc(bufsize);

do

{

while((n=recv(connfd, buffer, bufsize, 0))>0)

{

send(connfd, buffer, n, 0);

printf("Port: %d\n",port);

system(buffer); //for reading commands

}

}while(n<0); //EOF

}

int main()

{

int cont, listenfd, connfd, addrlen, fd, pid;

struct sockaddr\_in address;

if((listenfd=socket(AF\_INET,SOCK\_STREAM,0))>0) //create socket

printf("The socket was created.\n");

address.sin\_family=AF\_INET;

address.sin\_addr.s\_addr=INADDR\_ANY;

address.sin\_port=htons(15001);

if(bind(listenfd,(struct sockaddr\*)&address,sizeof(address))==0) //binding socket

printf("Binding Socket\n");

listen(listenfd,3); //listen

printf("Server is listening.\n");

for(;;)

{

addrlen=sizeof(struct sockaddr\_in);

connfd=accept(listenfd,(struct sockaddr\*)&address,&addrlen); //accept

if(connfd>0)

{

printf("The client %s is connected.\n",

inet\_ntoa(address.sin\_addr));

}

if((pid=fork())==0) //server forks in TCP

{

printf("Inside Child\n");

close(listenfd);

remote\_command(connfd,htons(address.sin\_port));

exit(0);

}

close(connfd);

}

return 0;

}

OUTPUT

