**Name:** Irfan Ullah Khan

**Student No:** 22010886

**Semester:** 1st

**Assignment No:** 05

**Department:** Computer Engineering (MSc)

**Subject:** Advanced Networking Programming

**Submitted To:** Prof. Devrim Seral

#include <sys/types.h>

#include <sys/socket.h>

#include <netinet/in.h>

#include "iniparser.h"

#include <netdb.h>

#include <stdio.h>

#define BUFFER\_SIZE 4096

void error(char \*msg)

{

perror(msg);

exit(0);

}

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

int Sockfd, PORT\_NUMBER, n;

struct sockaddr\_in serv\_addr;

struct hostent \*server;

char buffer[BUFFER\_SIZE];

if (argc < 3)

{

fprintf(stderr,"usage %s HOST\_NAME PORT\n", argv[0]);

exit(0);

}

PORT\_NUMBER = atoi(argv[2]);

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

if (Sockfd < 0)

error("Opening socket ERROR");

server = gethostbyname(argv[1]);

if (server == NULL)

{

fprintf(stderr,"error generated, because no such kind of host\n");

exit(0);}

#include <stdlib.h>

#include <errno.h>

#include <unistd.h>

#include <string.h>

#include <sys/wait.h>

#include <sys/socket.h>

#include <resolv.h>

#include <arpa/inet.h>

#include <pthread.h>

/\* Definations \*/

#define DEFAULT\_BUFLEN 1024

#define PORT 1886

void PANIC(char\* msg);

#define PANIC(msg) { perror(msg); exit(-1); }

/--------------------------------------------------------------------/

/--- Child - echo server ---/

/--------------------------------------------------------------------/

void\* Child(void\* arg)

{ char line[DEFAULT\_BUFLEN];

int bytes\_read;

int client = \*(int \*)arg;

do

{

bytes\_read = recv(client, line, sizeof(line), 0);

if (bytes\_read > 0) {

if ( (bytes\_read=send(client, line, bytes\_read, 0)) < 0 ) {

printf("Send failed\n");

break;

}

} else if (bytes\_read == 0 ) {

printf("Connection closed by client\n");

break;

} else {

printf("Connection has problem\n");

break;

}

} while (bytes\_read > 0);

close(client);

return arg;

}

/--------------------------------------------------------------------/

/--- main - setup server and await connections (no need to clean ---/

/--- up after terminated children. ---/

/--------------------------------------------------------------------/

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

{ int sd,opt,optval;

struct sockaddr\_in addr;

unsigned short port=0;

while ((opt = getopt(argc, argv, "p:")) != -1) {

switch (opt) {

case 'p':

port=atoi(optarg);

break;

}

}

if ( (sd = socket(PF\_INET, SOCK\_STREAM, 0)) < 0 )

PANIC("Socket");

addr.sin\_family = AF\_INET;

if ( port > 0 )

addr.sin\_port = htons(port);

else

addr.sin\_port = htons(PORT);

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

// set SO\_REUSEADDR on a socket to true (1):

optval = 1;

setsockopt(sd, SOL\_SOCKET, SO\_REUSEADDR, &optval, sizeof optval);

if ( bind(sd, (struct sockaddr\*)&addr, sizeof(addr)) != 0 )

PANIC("Bind");

if ( listen(sd, SOMAXCONN) != 0 )

PANIC("Listen");

printf("System ready on port %d\n",ntohs(addr.sin\_port));

while (1)

{

int client, addr\_size = sizeof(addr);

pthread\_t child;

client = accept(sd, (struct sockaddr\*)&addr, &addr\_size);

printf("Connected: %s:%d\n", inet\_ntoa(addr.sin\_addr), ntohs(addr.sin\_port));

if ( pthread\_create(&child, NULL, Child, &client) != 0 )

perror("Thread creation");

else

pthread\_detach(child); /\* disassociate from parent \*/

}

return 0;

}