


# National University of Computer and Emerging Sciences, Lahore Campus

	Course Name:	Computer Networks	Course Code:	CL-307
	Program:	BS(Computer Science)	Semester:	Fall 2020
	Duration:	2 hrs	Total Marks:	20
	Paper Date:	02-November-2020	Weightage	20%
	Section:	ALL	Page(s):	4
	Exam Type:	Mid Exam		

Student : Name: \_\_\_\_\_ Roll \_\_\_\_\_

Instruction/Notes: **READ ALL INSTRUCTIONS CAREFULLY.**

1. Understanding the question paper is also part of the exam, so do not ask any clarification. Make suitable ASSUMPTIONS.
2. Final Submissions should be done in your respective section folder on \\cactus\Xeon\Fall2020\COMPUTER NETWORKS LAB\Mid Submission. For question your server and client files should be named as **server.c** and **client.c** in a single zip file. Zip file must be renamed after your roll number and section e.g., "18L-1125-A". Multiple submissions are not allowed (if done, only latest one will be considered).
3. The second submission is on Slate in your respective section, follow the details: Slate>CN Lab >assignments>CN-Mid Exam, remaining formatting details are same.
4. Your cell phones/smart watches should be turned off and placed upside down.
5. It is your responsibility to protect your code and save it from being copied. If you don't protect it, all matching codes will be considered copy/cheating cases. **No leniency on plagiarism.**
6. Any kind of cheat sheet/code if found in your PC will result in immediate disqualification from Exam and 'F' as final grade in Computer Networks Lab. So make sure you delete everything from Desktop of your windows as well as Ubuntu. Also delete all the files permanently from Recycle Bin and Trash respectively for Windows and Ubuntu. Delete all files from your Z Drives before starting the exam.
7. You are immediately disqualified from the exam if:
  - i. You are seen talking, whispering, borrowing or looking at someone's PC
  - ii. A USB is found attached to your PC
  - iii. You are seen using cell phone/smart watch.
  - iv. You are caught accessing internet

## TCP SOCKET PROGRAMMING

(Marks: 30)

\*\*\*\*Submission: You have to submit your (Client.c) and (Server.c) files in a folder named your Roll#\*\*\*\*

**Note: Find relevant text files on Xeon & Slate**

### Online Bus Ticket Booking Application

Travelling gives us an opportunity to visit new places or reach our destinations. Being a resident of a different city for the purpose of education or job, we need to travel locally over buses more often. So, online bus ticket booking system can help us to book our ride sitting anywhere, despite of the worry of being there on bus terminal on time.

Your task is to write an **online bus ticket booking application**, in which a client/user sends a query of **schedule** to the server/online system and the server replies with the complete details of schedule after queering about city it wants to travel.. Basic working of this system is as follows:

- User sends a query to the server of **schedule of bus** (what are the timings for today). (1)
- Server queries back the client 'where do you want to travel, please mention city name' after client sends back the city name server sends one of its(city specific) text files to the client, finds that city file named '**city ABC**' maintained at the server side. These two (for two different cities) files are already provided to you, and you are not supposed to make any changes in these files. (4)
- The client then sends back the new txt file '**booking**' specifying the destination and time of departure he wants to travel. (2)
- If the server found this scheduled ride in '**city ABC**' file then book the ride and replied the client 'the booking is successful please note the booking details written in new txt file '**booking-final**' which includes the details about bus#, seat #, departure time from Lahore and the arrival time of destination city. (3)  
If the client requested scheduled ride is not found in '**city ABC**' txt file then server replies the user that "**schedule Not Found**". (2)
- After that client can either send another query or can send '-1' which would mean that client wants to disconnect from the server. (1)
- Upon receiving '-1' from the client side, server will reply with '**Disconnected successfully**' to client and then server will remove this thread from the pool of running threads. (2)

destination time did

2/  
4



You are to devise this application using **TCP\_Multithreaded** server-client program. Make sure that the server is always in the listening state and since multiple users can query the server at the same time so include TCP\_Multithreading in your code. Server can at max make connection with **three** users at a time. (5)

### Multithreading Socket Programming Syntax

- `int socket(int domain, int type, int protocol);`
  - `domain = AF_INET, AF_INET6`
  - `type = SOCK_STREAM, SOCK_DGRAM`
  - `protocol = 0(preferred), IPPROTO_TCP, IPPROTO_UDP, IPPROTO_ICMP`
- `int bind(int socket, struct sockaddr *name, int namelen)`
- `struct sockaddr_in {`
  - `short sin_family; // e.g. AF_INET, AF_INET6`
  - `unsigned short sin_port; // e.g. htons(3490)`
  - `struct in_addr sin_addr; // see struct in_addr, below`
  - `char sin_zero[8]; // zero this`
- `struct in_addr {`
  - `unsigned long s_addr; // load with inet_addr()`
- `struct sockaddr {`
  - `unsigned short sa_family; // address family, AF_XXX`
  - `char sa_data[14]; // 14 bytes of protocol address`
- `int listen(int socket, int backlog)`
- `int accept(int socket, struct sockaddr *addr, int *addrlen)`
- `int connect(int socket, struct sockaddr *addr, int addrlen)`
- `int send(int socket, const void *buf, int buflen, int flags);`
- `int recv(int socket, void *buf, int buflen, int flags);`
- `int sendto(int socket, const void *buf, int buflen, int flags, struct sockaddr* to, int tolen);`
- `int recvfrom(int socket, void *buf, int buflen, int flags, struct sockaddr* from, int *fromlen);`
- `int close(int socket)`

### Multithreading:

- `int pthread_create(pthread_t *thread, pthread_attr_t *attr, void *(*start_routine)(void *), void *arg);`
- `void pthread_exit(void *value_ptr);`
- `int pthread_join(pthread_t thread, void **value_ptr);`

### Filing Syntax:

- `ptr = fopen("filepath", "mode")`
- `int fgetc( FILE * fp ) //read single char from file`
- `fclose(fptr)`
- `type arrayName [ arraySize ]; // To declare an array`
- `int strcmp(const char *s1, const char *s2); // string compare`

### Headers File:

- `#include <stdio.h>`
- `#include <string.h>`
- `#include <sys/socket.h> //socket`
- `#include <arpa/inet.h> //inet_addr`
- `#include <stdbool.h>`
- `#include <pthread.h> //threads`