# **National University of Computer and Emerging Sciences LHR**

|  |  |  |  |
| --- | --- | --- | --- |
| Course | CN Lab | Course Code | CL3001 |
| Program | BS CS | Semester | Fall 2023 |
| Duration | 2 hr 30 min | Total Marks | 50 |
| Date | 25/Oct/2023 | Weightage | 25% |
| Section | 5J1 & J2 | Total Parts | 4 parts |
| Student Name | Abdullah Tahir | Student Roll # | 21L-5419 |
| Submission Link: (mandatory)Then submit on GCL also. | **\\cactus1\Xeon\Fall 2023\CN Lab Mid\5J Exam** **File NAME: ROLL#\_SEC\_NAME**  Submit all files, input files you use, compiled word file with answers and screenshots and .c files as well. | | |

# Rules:

• Understanding the question statement is also part of the exam, so **do not ask any questions** whatsoever. In case of any ambiguity, make suitable assumptions.

• You have to complete the exam in 2 hours and 30 minutes including the submissions, no late submissions will be acceptable whatsoever.

• Midterm exam **helping materials are placed on your Google classroom.**

• **Rename your submission folder to your Roll number in the format 20L-XXXX**, by putting your roll

number. Submission path is your google classroom. Folder has already been created for this purpose.

• Your code should be indented and commented properly. Use meaningful variable names. Failure to comply will result in marks deduction. **Submit screenshots where suitable**.

• Any kind of cheat sheet/code if found in your PC will result in immediate disqualification from midterm 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.

• **If any student is found browsing any website, his/her exam will be CANCELLED IMMEDIATELY.**

• INTERNET USAGE IS ONLY ALLOWED TO MAKE SUBMISSION ON GOOGLE CLASSROOM.

• It is your responsibility to save your code from being copied. All matching codes will be considered cheating cases. **PLAGIARISM will result in forwarding of case to Disciplinary Committee.**

• In case of missing or corrupted submission, ZERO marks will be awarded.

• 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.

# Objective: [1x10 Marks]

Instructions: Select and **BOLD+HIGHLIGHT** the answer part.

Example: This is a CN Lab mid?

**A Yes**

B No

1. Which layer of the OSI model is responsible for establishing, maintaining, and terminating connections between two network devices?

A. Network Layer

**B. Transport Layer**

C. Data Link Layer

D. Presentation Layer

2. What is the primary purpose of using Wireshark in network analysis and troubleshooting?

A. Configuring network devices

B. Analyzing network security

**C. Monitoring and capturing network traffic**

D. Creating network diagrams

3. In TCP socket programming, which function is used to listen for incoming connections on a server socket?

A. connect()

B. bind()

**C. listen()**

D. send()

4. What does the UDP in UDP socket programming stand for?

A. Universal Data Protocol

**B. User Datagram Protocol**

C. Unicast Data Packet

D. Underlying Data Parser

5. Which function is used to open a file in C for reading and writing, allowing both read and write operations on the file?

A. file\_open()

B. open\_file()

**C. fopen()**

D. create\_file()

6. In Wireshark, what is a "packet capture filter" used for?

A. To edit captured packets

**B. To prevent specific packets from being captured**

C. To organize captured packets into folders

D. To increase the capturing speed of packets

7. Which socket programming paradigm provides a connectionless, unreliable data transfer service?

A. TCP

B. HTTP

**C. UDP**

D. SMTP

8. What is the purpose of the fclose() function in C file management?

A. To create a new file

B. To flush file contents to disk

C. To open a file for reading

**D. To close a previously opened file**

9. In UDP socket programming, which function is used to receive data from a socket?

**A. recv()**

B. read()

C. sendto()

D. connect()

10. What is the role of a socket in networking and socket programming?

A. To store network configuration settings

B. To route packets within a network

**C. To provide an endpoint for communication**

D. To encrypt network traffic

# Tasks:

## Task 1: [20 marks]

Load the Session file **ICMP\_Session.** Now filter out all non-ICMP packets BY typing “icmp” (without quotes)**.**

Analyze the ICMP Packets and answer the following questions

1- Are ICMP messages sent over UDP or TCP?

|  |
| --- |
| **TCP**  C:\Users\lab\Desktop\1.PNG |

2- What is the link-layer (e.g., Ethernet) address of the host?

|  |
| --- |
| **192.168.100.1**  **C:\Users\lab\Desktop\2.PNG** |

3- Which kind of request is sent through these ICMP packets?

|  |
| --- |
| Echo request, Echo reply |

4- How many requests are sent through the host?

|  |
| --- |
| 4 request are sent by host |

5- What is the IP address of your host? What is the IP address of the destination host?

|  |
| --- |
| Source: 192.168.33.110  Destination: 172.217.27.36 |

6- Why is it that an ICMP packet does not have source and destination port numbers?

|  |
| --- |
| The ICMP packet does not have source and destination port numbers because it was designed to communicate network-layer information between hosts and routers, not between application layer processes. |

7- What values in the ICMP request message differentiate this message from the ICMP reply message?

|  |
| --- |
| The values that differentiate ICMP request messages from ICMP reply messages are the "Type" field (8 for request, 0 for reply) and the presence of an "Identifier" and "Sequence Number" in the ICMP data portion, which facilitate matching requests with corresponding replies. The first byte specifies the type of ICMP message. |

8- Examine one of the ping request packets sent by your host. What are the ICMP type and code numbers? What other fields does this ICMP packet have? How many bytes are the checksum, sequence number and identifier fields?

|  |
| --- |
| Type: 8 (Echo (ping) request)  Code: 0  Checksum: 0x4d39 [correct]  [Checksum Status: Good]  Identifier (BE): 1 (0x0001)  Identifier (LE): 256 (0x0100)  Sequence Number (BE): 34 (0x0022)  Sequence Number (LE): 8704 (0x2200) |

9- Examine the corresponding ping reply packet. What are the ICMP type and code numbers? What other fields does this ICMP packet have? How many bytes are the checksum, sequence number and identifier fields?

|  |
| --- |
| Type: 0 (Echo (ping) reply)  Code: 0  Checksum: 0x5539 [correct]  [Checksum Status: Good]  Identifier (BE): 1 (0x0001)  Identifier (LE): 256 (0x0100)  Sequence Number (BE): 34 (0x0022)  Sequence Number (LE): 8704 (0x2200) |

10-Examine the packet no 56. What are the ICMP type and code numbers? Why is the IP and TCP

Header included in the ICMP Header? What does these headers depict?

|  |
| --- |
| Type: 3 (Destination unreachable)  Code: 3 (Port unreachable)  The ICMP message includes the IP and TCP headers to provide diagnostic information about why the original packet couldn't be delivered. The IP header contains source and destination IP addresses, while the TCP header includes port information. This helps in troubleshooting network issues related to unreachable ports. |

## Task 2: [20 marks]

**Online Movie Ticketing system**

You are tasked with creating a basic online movie ticket booking system using a server-client program. The system should allow users to view a list of available movies and book tickets. Here's how the system should work:

1. The server displays a welcome message to clients and provides a list of available movies with unique identifiers.

In theaters:

[1] Donkey King

[2] Avengers

[3] Bohemian Rhapsody

[4] Halloween

[5] First Man

The client will select a movie by send movie number in reply along with number of ticket in following format.

2 4

i.e. (movie, tickets) this example means “book 4 tickets of movie 2 avengers”

1. The client selects a movie by sending the movie number along with the number of tickets they want to book in the following format: **2 4** (meaning they want to book 4 tickets for movie Avengers).
2. The server checks the availability of the selected movie and the number of seats. If there are not enough seats available, the server sends a relevant message.
3. If the reservation is successful, the server calculates the total bill (assuming a ticket price of 500 per ticket) and sends it to the client.

Your Total Bill for the show Avengers is 2000 Rs.

Press 'y' to proceed or press 'n' to cancel.

1. If the user presses 'y', the server decreases the number of tickets available for that movie and updates a text file, "movieTicketsInfo.txt," if necessary. If there are no more tickets for a show, the server removes that show entry from the file.
2. After successfully updating the ticket information on the server side, the server prompts the user to enter their CNIC (assuming a 4-digit number), movie number, and the number of reserved tickets.
3. The server saves this booking information in a file named "BookingRecord.txt."
4. Finally, the server sends a message to the client, indicating whether the reservation was successful ("Reservation Done") or if any errors occurred ("Reservation Cancel").

Your task is to implement this simplified movie ticket booking system using a server-client program

Here are snapshots of the files:

|  |  |
| --- | --- |
| **movieTicketsInfo.txt**  5  1/Donkey King/500  2/Avengers/300  3/Bohemian Rhapsody/300  4/Halloween/400  5/First Man/500 | **BookingRecord.txt**  4  1990/2/4  6667/4/4  7865/4/6  5849/2/3 |

In BookingRecord.txt first line tells count of users. (Format: CNIC/Movie number/tickets)

In movieTicketsInfo.txt first line tells count of shows. (Format: Movie number/Name/Tickets)

**Make txt files at your end with these contents before starting the question.**

|  |
| --- |
| **CLIENT:**  #include <stdio.h>  #include <string.h>  #include <sys/socket.h>  #include <arpa/inet.h> #include <stdlib.h>  int main(void)  {  int socket\_desc;  struct sockaddr\_in server\_addr;  char server\_message[2000], client\_message[2000];  // ... (Initialization and socket creation code remains the same)  // Input movie selection and ticket count  int movieSelection, ticketCount;  printf("Enter the movie number and the number of tickets (e.g., 2 4 for Avengers): ");  scanf("%d %d", &movieSelection, &ticketCount);  sprintf(client\_message, "%d %d", movieSelection, ticketCount);  // Send movie selection and ticket count to the server  send(socket\_desc, client\_message, strlen(client\_message), 0);  // Receive and display the server's response  recv(socket\_desc, server\_message, sizeof(server\_message), 0);  printf("Server Message: %s\n", server\_message);  // ... (Cleanup and socket closure code remains the same)  return 0;  }  **SERVER:**  #include <stdio.h>  #include <string.h>  #include <sys/socket.h>  #include <arpa/inet.h>  // Function to update ticket information in the file  void updateTicketInfo(int movieSelection, int ticketCount)  {  char fileName[] = "movieTicketsInfo.txt";  FILE\* file = fopen(fileName, "r+");  if (file == NULL)  {  printf("Failed to open the ticket information file.\n");  return;  }  int availableSeats[5] = {10, 5, 8, 6, 7};  if (availableSeats[movieSelection - 1] >= ticketCount)  {  availableSeats[movieSelection - 1] -= ticketCount;  fseek(file, 0, SEEK\_SET);  fprintf(file, "%d %d %d %d %d", availableSeats[0], availableSeats[1], availableSeats[2], availableSeats[3], availableSeats[4]);  }  else  {  printf("Not enough seats available for the selected movie.\n");  }  fclose(file);  }  int main(void)  {  int socket\_desc, client\_sock, client\_size;  struct sockaddr\_in server\_addr, client\_addr;  char server\_message[2000], client\_message[2000];  // ... (Initialization and socket creation code remains the same)  // Display a welcome message and available movies  strcpy(server\_message, "Welcome to Movie Ticket Booking. Available Movies:\n");  char movies[5][50] = {  "1. Avengers: Endgame (10 seats available)",  "2. The Shawshank Redemption (5 seats available)",  "3. The Godfather (8 seats available)",  "4. Inception (6 seats available)",  "5. Pulp Fiction (7 seats available)"  };  for (int i = 0; i < 5; i++)  {  strcat(server\_message, movies[i]);  strcat(server\_message, "\n");  }  send(client\_sock, server\_message, strlen(server\_message), 0);  // Receive the client's movie selection and ticket count  recv(client\_sock, client\_message, sizeof(client\_message), 0);  int movieSelection, ticketCount;  sscanf(client\_message, "%d %d", &movieSelection, &ticketCount);  // Check movie availability and calculate the total bill  if (movieSelection >= 1 && movieSelection <= 5)  {  updateTicketInfo(movieSelection, ticketCount);  int totalBill = 500 \* ticketCount;  sprintf(server\_message, "Reservation successful. Total bill: %d", totalBill);  // Prompt the user for CNIC, movie number, and number of reserved tickets  int cnic, reservedMovie, reservedTickets;  printf("Enter your 4-digit CNIC: ");  scanf("%d", &cnic);  reservedMovie = movieSelection;  reservedTickets = ticketCount;  // Save booking information in "BookingRecord.txt"  FILE\* bookingFile = fopen("BookingRecord.txt", "a");  if (bookingFile != NULL)  {  fprintf(bookingFile, "CNIC: %04d, Movie: %d, Tickets: %d\n", cnic, reservedMovie, reservedTickets);  fclose(bookingFile);  }  else  {  printf("Failed to record the booking.\n");  }  }  else  {  strcpy(server\_message, "Invalid movie selection.");  }  send(client\_sock, server\_message, strlen(server\_message), 0);  // ... (Cleanup and socket closure code remains the same)  return 0;  } |

