CS 134 Final Examination

Fall 2006

This is a closed book exam. You have 150 minutes to complete the exam. There are five questions on this examination. The point values for the questions are shown in the table below. Your answers should fit in the space provided in the exam booklet. Paper for scrap work will be made available during the examination.

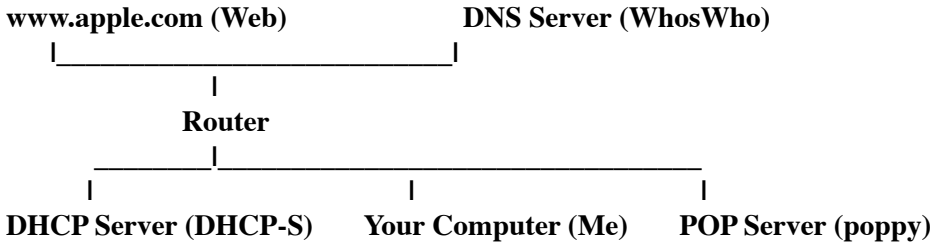
Good Luck!

Question	Point	Score
1	20	
2	20	
3	20	
4	20	
5	20	
TOTAL	100	

NAME: I have neither given nor received aid on this examination.

_____ Signature: _____

1) When a computer is connected to a network, it must send and receive many packets before it is able to perform "useful" work like browsing the web. In this problem, we want you to describe the network processes that occur between when you plug your computer into an Ethernet jack and when a web page (e.g., www.apple.com) is first visible on the screen, in the network pictured below:



Complete the timeline on the following page, by briefly describing the details of each Ethernet packet transmitted. The packets should be listed in the order in which they would be transmitted. By "transmitted," we mean "put on the wire."--note that the same *IP packet* may be put on the wire more than once in different *Ethernet packets* if it is forwarded by a router.

You may use the simplified versions of protocols that we described in class; e.g., although DHCP actually transmits *four* types of packets, the basic request/response that we showed in class is sufficient for this question. In the interests of brevity, please DO NOT show TCP SYN, TCP FIN, or TCP ACK packets. We have filled in some boxes for you as examples.

For this problem, assume that:

- All networks are Ethernets
- There are no Ethernet collisions
- No packets are lost, corrupted, or dropped in transmission
- The web page can be fetched with a single HTTP request (e.g., no images, favicon.ico, etc.)
- Local network does not use a NAT
- Local network does not use authentication
- All traffic is between the machines shown in the diagram
- Each message fits in a single UDP, TCP, IP, or Ethernet packet

We recommend that you leave empty rows between packets to leave yourself room to insert more packets later if you discover an omission in your solution. We have also provided an additional blank copy of the table for scratch work. Please clearly indicate which copy you would like graded. For your reference, a list of all of the protocols discussed in class is below.

TCP ARP UDP IP Ethernet OSPF POP DNS HTTP TOC FLAP DHCP

High-Level Protocol	Ethernet From Addr	Ethernet To Addr	Conceptual Description of Contents
DHCP	Me	All of Local Net	"My hardware address is (MAC address)...what is my IP address?"
DHCP	DHCP-S	Me	• My IP address • The router's IP address • A DNS server's IP address
ARP	Me		
HTTP		Me	<HTML><HEAD> .... web page text

High-Level Protocol	Ethernet From Addr	Ethernet To Addr	Conceptual Description of Contents
DHCP	Me	All of Local Net	"My hardware address is (MAC address)...what is my IP address?"
DHCP	DHCP-S	Me	• My IP address • The router's IP address • A DNS server's IP address
ARP	Me		
HTTP		Me	<HTML><HEAD> .... web page text

2) Use Djikstra's Algorithm to complete the following table using **C** as the starting point.

Dijkstra's algorithm

Mark starting column done
For each neighbor of the starting position
    set neighbor's min path length equal to cost of link from start to neighbor
    set neighbor's best first step equal to itself.

while there are still columns that are not marked as done:
    Find the not-done column with the smallest min path length and mark it as done
    For each neighbor of the column just marked done
        if cost of link to neighbor plus this column's min length < neighbor's min length then
            set neighbor's min length to cost of link plus this column's min length, &
            set neighbor's first step to this column's first step.

NOTE!!!! WE WANT YOU TO COMPUTE THE BEST PATHS FROM **C** TO OTHER NODES, NOT THE BEST PATHS FROM **A**.

	A	B	C	D	E	F	G	H
Min path len			0					
Best 1st step			-					
Done?								
Connections to neighboring routers	B - 4	A - 4	B - 12	A - 2	B - 4	C - 2	C - 7	G - 2
	D - 3	E - 4	F - 2	E - 5	D - 5	E - 5	F - 10	
		C - 12	G - 7		F - 5	G - 4	H - 2	

Just in case you need room for scratch work on this problem, we have included extra copies of the table on the following page. Please clearly indicate which copy we should grade!

	A	B	C	D	E	F	G	H
Min path len			0					
Best 1st step			-					
Done?								
Connections to neighboring routers	B - 4	A - 4	B - 12	A - 2	B - 4	C - 2	C - 7	G - 2
	D - 3	E - 4	F - 2	E - 5	D - 5	E - 5	F - 10	
		C - 12	G - 7		F - 5	G - 4	H - 2	

	A	B	C	D	E	F	G	H
Min path len			0					
Best 1st step			-					
Done?								
Connections to neighboring routers	B - 4	A - 4	B - 12	A - 2	B - 4	C - 2	C - 7	G - 2
	D - 3	E - 4	F - 2	E - 5	D - 5	E - 5	F - 10	
		C - 12	G - 7		F - 5	G - 4	H - 2	

3) Consider the application of two dimensional parity bits to the transmission of very small units of data --- four bits. For example, to transmit the message 1101 we would form the table:

1	1
0	1

and then add parity bits to obtain:

1	1	0
0	1	1
1	0	1

and would finally transmit the binary sequence 110011101 to deliver the data reliably through the network.

a) Suppose when using this scheme, you receive the message 110100000. Assuming at most one bit was damaged, what can you say about the four bits the sender intended to transmit to you? Briefly explain your answer.

b) Suppose when using this scheme, you receive the message 000011011. Assuming at most one bit was damaged, what can you say about the four bits the sender intended to transmit to you? Briefly explain your answer.

c) Using two-dimensional parity with just 4 data bits is not very efficient. There are more parity bits in the messages transmitted than actual data bits! One way one might try to make it more efficient is to simply not send the last bit. The last bit is, after all, just a parity bit for parity bits. It isn't clear whether such a bit really has any value.

So, suppose someone sends you data by computing two-dimensional parity bits as described above but only sending you the first 8 of the 9 bits that would normally be sent. Assuming at most one bit is damaged, if you receive the 8-bit message 00000001, what can you say about the four bits the sender intended to transmit to you? Briefly explain your answer.

d) In part (c) we had you consider how failing to send one bit of a table of data with two-dimensional parity bits added would impact the receiver's ability to correct one bit errors. Now, we would like you to consider the impact this change has when two bit errors occur.

When all 9 bits of a two-dimensional parity table are sent, it is always possible to identify a 2 bit error as an error that cannot be corrected. If the 9th bit is not sent, however, there are 4 distinct patterns of pairs of damaged bits that result in parity mismatch patterns identical to those associated with certain 1 bit errors. As a result, the receiver of a message containing a pair of such damaged bits might incorrectly conclude that a single bit was damaged and incorrectly "fix" the error.

Show an example of one such error.

e) In part (e), we mentioned that there are 4 patterns of two bit errors that might lead a receiver to incorrectly believe it was possible to fix the damage. If the probability of a bit being damaged in transmission is p independent of the other bits in a message, what is the probability that such a two bit error will occur?

4) You may recall that way back during your second week in lab there was something a bit odd about the code you had to write to implement a POP client. Among all of the other instructions we gave you for that lab, we included a summary of how POP worked which, among other things, told you "When you are all done you will send the server a simple "QUIT" command and then close the connection." To most beginners, this process seems a bit redundant. Knowing little about TCP, it can be a bit hard to understand how or why sending the "QUIT" command and closing the connection are both necessary. Each one of them in some way tells the server that the client is done. Why are both necessary. To many students, performing both steps seems redundant.

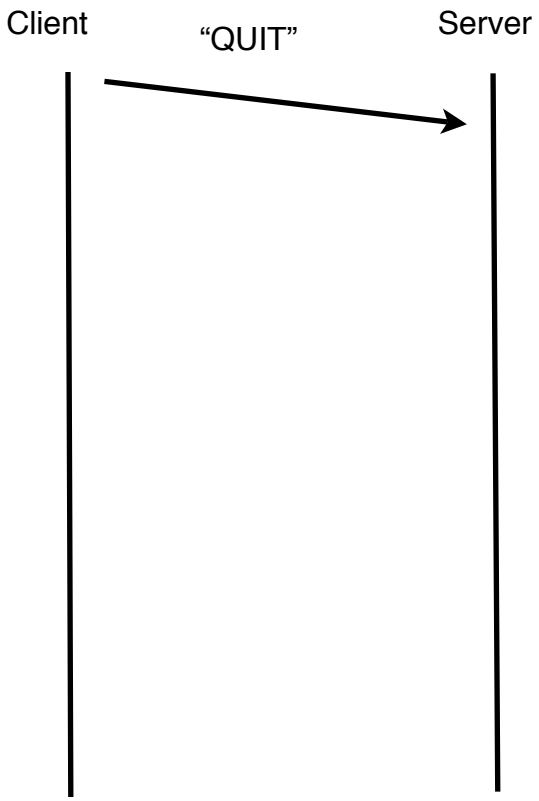
Now, of course, being experts on TCP, you clearly understand the difference between sending the "QUIT" command and closing the connection. To demonstrate this, we would like you to answer the following questions in the context of a Server whose dataAvailable method contains the code:

```
if (lineFromClient.equals("QUIT") {
        toClient.out.println("+OK");
        toClient.close();
}
```

a) Describe the sequence of TCP packets that would be exchanged between a POP client and a POP server if the client program executed just an instruction like

```
toServer.out.println( "QUIT" );
```

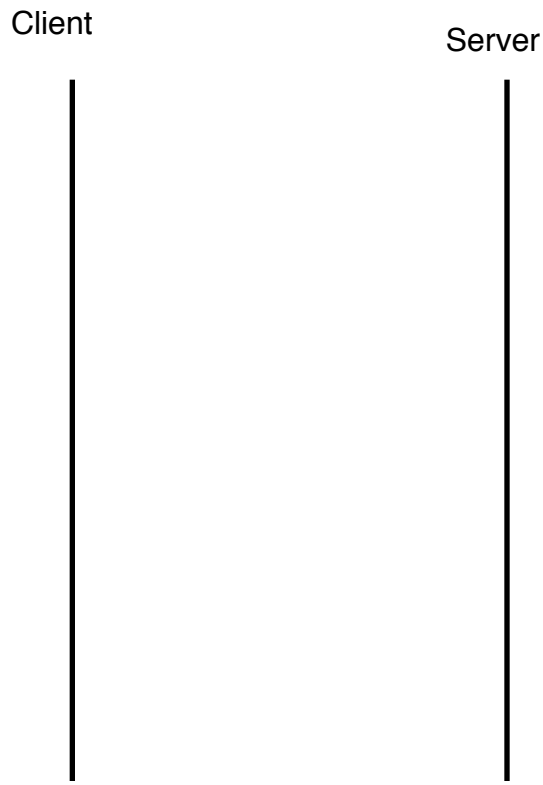
to send a "QUIT" command to the server. Assume in this and the following steps that all packets are delivered correctly so that no retransmissions are required, and that all ACKs are in their own TCP packets (not combined with some other data).



b) Describe the sequence of packets that would be exchanged between a POP client and a POP server if a client program executed just an instruction like

```
toServer.close();
```

to close the underlying TCP connection.



c) Describe what would happen if the program performed the required steps in the wrong order by executing the instructions

```
toServer.close();
toServer.out.println( “QUIT” );
```

5) Suppose that a collection of email messages are stored in an array using an EmailMessage class like the one that you used in your test program to represent the individual messages. That is, you have a class whose declaration looks like:

```
public class MessageCollection {
        ....
        private EmailMessage [] allMessages;
        ...
        // lots of other interesting instance variables

        // some sort of constructor

        public EmailMessage getLargest() {
                ...
        }

        // lots of other interesting methods
        ...
}
```

and that the name allMessages is associated with an array containing all the email messages stored in someone's account by the code in the unseen constructor.

a) We would like you to provide the code for the method named getLargest. This method should return the EmailMessage whose text is the longest. Assume that the EmailMessage class provides a method named getLength that returns the length of the text of the entire message. In writing this method, you may assume that the array contains at least one element.

Show your code on the next page.

b) We changed our minds. This happens a lot in software development. Suppose instead that a collection of email messages are stored in a linked list using an EmailMessage class like the one that you used in your test program to represent the individual messages. That is, you have a class whose declaration looks like:

```
public class MessageList {

    private EmailMessage firstMessage;
    private MessageList restOfMessages;
    private boolean empty;
    ...
    // lots of other interesting instance variables

    // some constructors
    ...

    public EmailMessage getLargest() {
        ...
    }
    private boolean isLast() {
        return restOfMessages.isEmpty();
    }
    public boolean isEmpty() {
        return empty;
    }

    // lots of other interesting methods
    ...
}
```

We would like you to provide the code for the method named getLargest. This method should return the EmailMessage whose text is the longest. Assume that the EmailMessage class provides a method named getLength that returns the length of the text of the entire message. In writing this method, you may assume that it will never be called from outside the class on a collection unless that collection contains at least one EmailMessage.

Hint: Note that the MessageList class has a method isLast() that returns true when invoked on a list containing exactly one EmailMessage. This method will be more useful than the empty variable or the isEmpty method for implementing getLargest.

Show your code on the next page.