CS2105
Tutorial 3

Question 1

Launch your browser and open its network diagnostic tool. Then click the "Network" tab to observe network communication.

Copy-and-paste the following URL in the address bar of your browser:

http://tiny.cc/atupaz

Enter your choice and press the "Submit" button.

a) Look at the entry named "formResponse". What is the HTTP request method issued?

- POST

b) Briefly explain when HTTP POST and GET methods are used.

- The POST request method requests that a web server accepts and stores the data enclosed in the body of the request message. It is often used when uploading a file or submitting a completed web form.
- In contrast, the HTTP GET request method is designed to retrieve information from the server.

Question 2

Suppose that your department has a **local DNS server** for all computers in the department. You are an **ordinary user** (i.e., not a network/system administrator). Can you determine if an external Web site was likely accessed from a computer in your department a couple of seconds ago? Explain.

Question 2

Suppose that your department has a **local DNS server** for all computers in the department. You are an **ordinary user** (i.e., not a network/system administrator). Can you determine if an external Web site was likely accessed from a computer in your department a couple of seconds ago? Explain.

If IP address of this Web page has been queried by another computer seconds ago, your local DNS server should keep this knowledge in local DNS cache and is able to answer your query quickly. Otherwise, the query time will be long.

Question 3

You are given 4 programs: **TCPEchoServer.py**, **TCPEchoClient.py**, **UDPEchoServer.py** and **UDPEchoClient.py**.

a) Suppose you run **TCPEchoClient** before you run **TCPEchoServer**. What happens? Why?

When creating a local socket, client attempts to make a TCP connection to a non-existent server process. Exception will be thrown.
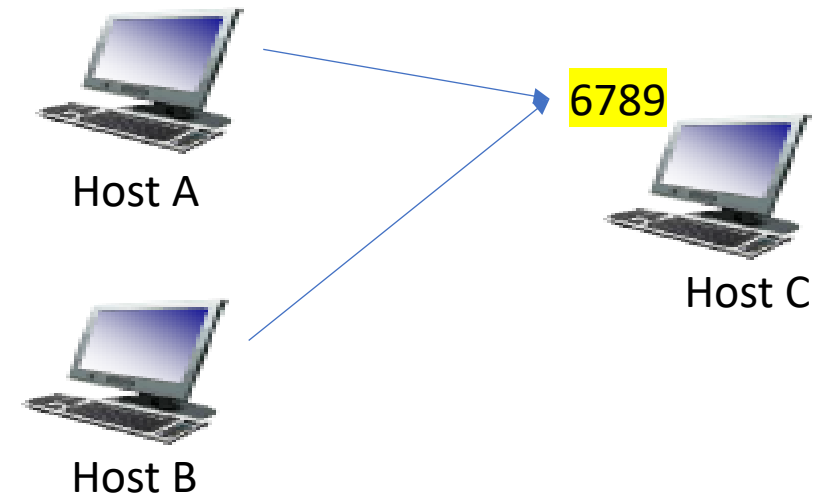
Question 3

You are given 4 programs: **TCPEchoServer.py**, **TCPEchoClient.py**, **UDPEchoServer.py** and **UDPEchoClient.py**.

b) Suppose you run **UDPEchoClient** before you run **UDPEchoServer**. What happens? Why?

UDP client doesn't establish connection to server when creating local socket. Thus, it works fine if you start client program first and then server program (but data sent to server are all lost).

Question 4

- Suppose a process in Host C has a UDP socket with port number 6789.

- Suppose both **Host A and Host B** each sends a **UDP segment** to Host C with destination **port number 6789**.

Host A

6789

Host C

Host B

Question 4

- Will both of these segments be directed to the same socket at Host C?
  - Yes, both segments will be directed to the same socket at C.

- If so, how will the process at Host C know that these two segments originated from two different hosts?
  - From the source IP address.
  - In python, recvfrom() function return client address
  - In java, DatagramPacket class provides a method getAddress() for us to learn the IP address of the origin of a packet.

Question 5

a) Suppose you have the following 2 bytes: **01011100** and **01100101**. What is the 1's complement of the sum of these 2 bytes?

rules: 0+0 =0

0+1 =1

1+1= 10 (sum = 0, carry = 1)

Sum: 11000001

Checksum: 00111110

$$0\ 1\ 0\ 1\ 1\ 1\ 0\ 0$$
$$0\ 1\ 1\ 0\ 0\ 1\ 0\ 1$$

Question 5

b) Suppose you have the following 2 bytes: **11011010** and **01100101**. What is the 1's complement of the sum of these 2 bytes?

1 1 0 1 1 0 1 0
0 1 1 0 0 1 0 1
_____

Sum: 01000000

Checksum: 10111111

Question 6

Suppose that UDP receiver computes the checksum for the received UDP segment and finds that it matches the value carried in the checksum field. **Can the receiver be absolutely certain that no bit errors have occurred?** You may use Q5 as an example to explain.

01011100
01100101
Checksum => 00111110

01111100
01000101
Checksum => 00111110

If sender transmits the following two bytes: 01011100 and 01100101, and the two bits highlighted in blue flip, then checksum remains unchanged and receiver will fail to detect this error.

Question 7

In our `rdt` protocols, why did we need to introduce sequence numbers?

Sequence numbers are required for a receiver to find out whether an arriving packet contains new data or is a retransmission.