Question 1: General Networking Concepts

(a) True or False

(i) Link bandwidth is usually measured in milliseconds.

- false
- measured in bits per second

(ii) HTTP is an example of a request-response protocol.

- true

(iii) DSL typically offers asymmetric bandwidth.

- true

(iv) A "bottleneck" link constrains the achievable end-to-end throughput.

- true

(v) The OSI network architecture is based on a 6-layer model.

- false
- based on a 7-layer model

(b)

- web browsing -> HTTP
- email access -> IMAP
- file sharing -> BitTorrent
- reliable transport -> TCP
- name resolution -> DNS

Question 2: Networking Fundamentals

(a)

- network structure diagram showing global ISP, regional ISP, home network, etc.
- the routers make up the core
- the applications and hosts (desktops, laptops, servers, phones) make up the edge

(b)

- assume processing delay negligible
- assume no queueing delay
- transmission delay is L / R where L is the packet length and R is the link bandwidth
    - = 500 kB / 1 Mbps
    - = 500 * 8 kb / 1000 kbps = 4000 s
- propagation delay is D / S where D is the distance and S is the propagation speed
    - = 386,000 km / 300,000 km/s ~= 1.3 s
    - RTT = 2.6 s
- total time is 4000 + 1.3 + 2(2.6) = 4,006.5 s

Question 3: Application Layer

(a) Copy diagram and explain DNS resolution in iterated and recursive modes.

Iterated mode:
- client contacts local name server looking for www.abc.com
    - local name server contacts root name server looking for www.abc.com
    - root name server returns IP of .com name server
    - local name server contacts .com name server looking for www.abc.com
    - .com name server returns IP of abc.com name server
    - local name server contacts abc.com name server looking for www.abc.com

- – abc.com name server returns IP for www.abc.com
- local name server returns IP for www.abc.com

Recursive mode:

- client contacts local name server looking for www.abc.com
  - – local name server contacts root name server looking for www.abc.com
    - * root name server contacts .com name server looking for www.abc.com
      - · .com name server contacts abc.com name server looking for www.abc.com
      - · abc.com name server returns IP for www.abc.com (to .com name server)
    - * .com name server returns IP for www.abc.com
  - – root name server returns IP for www.abc.com
- local name server returns IP for www.abc.com

(b)

Using a labeled diagram, show the basic operation of a playout buffer in a video streaming client.

(diagram omitted)

- playout buffer fills to a certain level at the beginning before playback of the video starts
- once playback begins, it drains data from the buffer at a constant rate
- the buffer is filled from the network at a variable rate due to network delays
- when the fill rate is lower than the drain rate, the buffer gradually empties
- when the fill rate is higher than the drain rate, the buffer gradually fills

When streaming video it sometimes happens that the playback stalls because the local client buffer is empty. Briefly describe two ways that stalls can be avoided?

- using a larger buffer and a longer playout delay at the beginning
  - – then larger fluctuations from the network can be tolerated
- streaming lower quality video from the source
  - – then the same amount of data covers more video time

– decreases buffer drain rate

– if using DASH, this can be done in the middle of playback

Question 4: Transport Layer

(a)

Explain the purpose of the port field in Internet transport protocol headers. Illustrate using an example.

- port field is used for demultiplexing
- when a computer receives a packet, it uses the destination port address (with the source IP address and the source port number, if this is connection-oriented demux) to tell which socket it should send the packet to – this socket is connected to a process, which receives the data and deals with it

Justify why an application programmer might decide to use UDP rather than TCP? Consider all performance-related factors that impact this decision.

UDP is unreliable data transfer – if packets get lost, they aren't retransmitted. TCP, on the other hand, guarantees that the packets will arrive (using retransmission and other techniques).

However, the techniques TCP uses to guarantee that packets will arrive generates overhead which reduces the available throughput. Also, in TCP there's handshaking required at the start of communication to create a connection, which reduces throughput in the same way. UDP doesn't have this overhead.

UDP also doesn't require connection state at the sender and the receiver, and so might be useful for embedded systems that don't have much memory available.

UDP has a smaller segment header, increasing throughput further.

UDP has no congestion control, which means an application using UDP can send as fast as it likes. This might be useful if an application needs to be able to send a lot of data in a small window (maybe for power efficiency?).

UDP can also be used in conjunction with reliable data transfer algorithms implemented at the application layer, which can allow application-specific error recovery.

The main motivation for an application to use UDP is the increased throughput. However, the application needs to be able to tolerate loss, as it's not guaranteed that all packets will get through.

(b)

Question

TCP uses a sliding-window protocol. Consider two hosts, A and B, with an open TCP session. A sends a segment with sequence number 2600 and after some time receives a segment from B with sequence number 22500 and acknowledgment number 3600 and Receive/Advertised Window set to 10000.

(i) How many bytes were received and confirmed by host B?

3600 - 2600 = 1000 bytes

- 3600 as an acknowledgements means "The next byte I'm expecting from you is number 3600"

    - 2600 as a sequence number means "The first byte in this segment is number 2600"
    - so bytes 2600 through 3599 were received, which is 3600 - 2600 = 1000 bytes

(ii) Is there a relationship between the values of the two sequence numbers? Explain your answer.

No, the two hosts pick random initial sequence numbers.

There's no need for a relationship between the sequence numbers. Host A can use B's numbering system for B's segments (e.g. when sending ACKs) and host B can use A's numbering system.

(iii) How will host A interpret the value of the Receive/Advertised Window field?

- A won't let any more than 10000 bytes be outstanding – if that many bytes are un-ACKed, it won't send any more segments.
- Could maybe also say A will reduce its sending window size to 10000 bytes?