DAT110 Exam
Candidate: 10

Task 1
a) All, 1,2,3,4,5.
b) 1.
c) 3.
d) 1.
e) 3.
f) 2.
g) b.
h) 1.
i) 2.
j) 1.

Task 2

a) In the broker we implemented:
   Dispatcher: Processes incoming messages on connections.
   Broker: A process continuously running which accepts incoming sessions from clients.
   Messaging server: Accepts new message connections from clients.
   Storage: Manages information stored of connected clients and the subscribtions they have.

b) The messaging layer provided the service of communication between client and broker. The information needed for the messaging client was server address:port.

c) Stobable-thread works by creating an abstraction of which starts, runs stops and cleans up a thread. This provides us with a way to automaticly call a method in a loop and a way to stop and clean up when the method is stopped.

d) The broker identifies clients where the message should be send by

e) The broker could be extended to be able to buffer messages and resend them later to clients subscribed to a topic but not currently connected to the broker by modifying it such that we can perform which would allow the client to resend them

Task 3

a) R = 10Mbps
Distance R1 → R2 = 10.000 meters
Signal propagation speed = 2 * 10^8 m/s = 200000000m/s
Packet L = 100.000 bits

TD = L/R = 100.000/10Mbps = 10ms

b) Propagation delay = distance/speed = 10000 / 2*10^8 = 0.05ms
Nodal delay = Processing delay + Queuing delay + Transmission delay + Propagation delay

Nd = 2ms + 10ms + 10ms + 0.05ms = 20.25ms.
The nodal delay is 20.25ms.

c) H1 → H3
Segment H1-R1
Qd: 10ms
Procd: 2ms
Td:10ms
Propd: 0.05ms

Segment R1-R2
Qd: 10ms
Procd: 2ms
Td: 10ms
Propd: 0.05ms

Segment R2-H3
Qd: 10ms
Procd: 2ms
Td:10ms
Propd: 0.05ms

Total delay: 60.75ms

d) The bottleneck in this case will be R1 → R2 since
R1→R2 100.000/10Mbps = 10ms
H1/H2 → R1 100.000/100Mps = 1ms

Task 4

a) The UDP protocol handles datagrams and provides checksums for checking data integrity while also providing port numbers which enables the addressing of different functions on the source and destination of the datagrams.

b) Source port: Identifies the port of the sender of the datagram.
Destination port: Identifies the destination/receivers port.
Length: The length in bytes of header and data of the datagram.
Checksum: Field for enabling the error checking of the header and data.
Application data: The payload sent by the sender of the datagram.

c) An application layer protocol that relies on UDP is VoIP. UDP is used in this application layer protocol since UDP is connectionless, data packets are sendt without negotiation and warning. While UDP does not lacks error control, the VoIP service does not require a completely reliable transport layer protocol for the experience of the user to be adequate. Packet loss often goes unnoticed since the audio quality does not degrade noticeably.

d) The network layer is able to deliver an UDP segment to a host delivered to the receiver with the specified port of the receiving application.

Task 5

a) The least cost path from R1 → R2 is:
R2 → R1 = 7
R2 → R3 → R1 = 1 + 2 = 3
The least cost path from R1 → R2 = R2 → R3 → R1 = 3.

b)

Y	Dr1(y)
R1	0
R2	7
R3	2
R4	inf

Y	Dr2(y)
R1	7
R2	0
R3	3
R4	inf

c) The distance vector Dr1 will be

Y	Dr1(y)
R1	0
R2	3
R3	2
R4	inf

Y	Dr1(y)
R1	0
R2	3
R3	2
R4	7

d)

Y	Dr2(y)
R1	3
R2	0
R3	1
R4	6

e. In order for R1 to know about the shorter route to R4, we would need two iterations, one for R2 to find and share its data, and one for R4 to share its data.

Task 6

a) The dynamic host configuration protocol (DHCP) can be used to automaticly obtain an IP address for H0.

b) Since the IP datagram is encapsulated into a frame and sent from H0 to H3, the IP address specified in the ARP query needs to be 108.201.1.3 corresponding to the R3.

c) The content of the switch table after the IP datagram has been sent from H0 to H3 is:

   Since H0 sent a an arp query + datagram to the switch which sent it through port 2 to the receiving address 3B:3B:3C:3D:3E:3F and the arp response with address A0:B0:C0:D0:E0:F0 has been sent into port 1. The switch table contains the following after H0 has sent an IP datagram to H3.

   A0:B0:C0:D0:E0:F0 → Port 1
   3B:3B:3C:3D:3E:3F → Port 2

d) The data will be encapsulated at H0, R3, R1, R2, 4 times.
   The data will be decapsulated at R3, R1, R2, H3, 4 times.
   Switch???

e) The IP address of 221.198.1.3 corresponds to the 32 bit address of 11011101.11000110.00000001.00000011.

f) **221.198.1.3 = 11011101.11000110.00000001.00000011**
   The router R1 will forward the address at interface 2 since
   Interface 2 = 11011101.11000110.00000001
   221.198.1.3 = 11011101.11000110.00000001.00000011
   Interface 2 is the closest match.

   132.200.1.1 = 10000100.11001000.00000001.00000001
   The router R1 will forward the address at interface 1 since
   Interface 1 = 10000100.11001000.00000001
   132.200.1.1 = 10000100.11001000.00000001.00000001
   Interface 1 is the closest match.

   108.201.1.5 = 01101100.11001001.00000001.00000101
   The router R1 will forward the address at interface 3 since
   Interface 3 = 01101100.11001001.00000001
   108.201.1.5 = 01101100.11001001.00000001.00000101
   Interface 3 I s the closest match.

g) 221.198.1/24 = 11011101.11000110.00000001.XXXXXXXX
The IP addresses available range from the lowest address
221.198.1.0 = 11011101.11000110.00000001.00000000 to the highest address at
221.198.1.255 = 11011101.11000110.00000001.11111111.

Since the addresses are can be described with the 24-bit address prefix
221.198.1.0/24 and routing is based on longest matching we can replace the two
entries for H3 and H4 with:
Destination: 11011101.11000110.00000001
Next hop interface: 2

Task 7

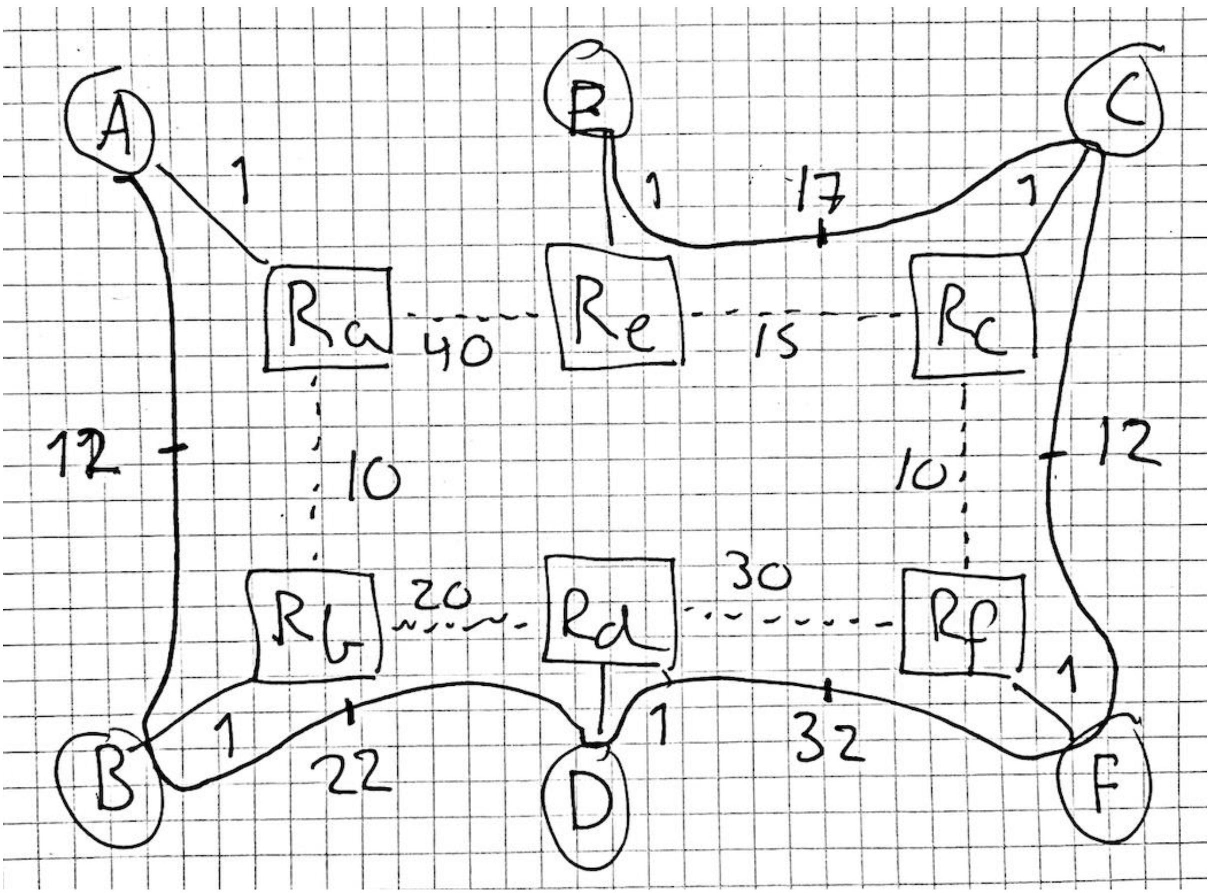
a) Asynchronous RPC is different from traditional RPC in the sense that asynchronous rpc separates the the remove procedure call from the return value by assigning a reply listener class which handles replies. While in a traditional rpc a client is blocked in a rpc until the call is returned. In the asynchronous rpc we are able to resolve issues of the traditional rpc in that we are able to avoid the blocking caused by the client/server waiting for the return call. We are also able to have multiple calls from a single client, handle slow clients which supply additional data and transferring larger amounts of data where without blocking the server or client from performing other tasks.

b) The key design issues we have to consider when designing servers are:

Scalability
Openness
Heterogenity
Security
Error handling
Concurrenc
Transparency.

Task 8

a).
A → B = 12
B → D = 22
D → F = 32
F → C = 12
C → E = 17



B)

Tree 1 = A – C – F – E – D – B
A – Ra – Re – Rc – C – Rf – F – Rc – Re - E – Rc – Rf – Rd – D – Rb - B
1+40+15+1+1+10+1+1+10+15+1+1+15+10+30+1+1+20+1 = 175
The overlay network path delay from A-B when we use tree one is 175.

c)
Tree 2 = B – A – E – D – F – C

Physical = C – Rc – Rf – Rd – Rb – B
= 1 + 10 + 30 + 20 + 1 = 52

Overlay = B – Rb – Ra - A – Ra – Re – E – Re – Rc – Rf – Rd – D – Rd – Rf – F – Rf – Rc – C
= 1+10+1+1+40+11+15+10+30+1+1+30+1+1+10+1 = 164

Message path	Delay		
	Tree 1		
	Overlay	Physical	RDP
C → B	164	52	1.32

Task 9

a)

b. P1 = P0,
P2 = P1
P3 = P2
P4 = P1

The possible values
R(x)? = are a,b,c,d.

c) Replication can be used for each node in the distributed system having their own copy which they can read at will if one other part of the system locks the original for a write operation. This provides increased performance and reduces chances of locking.

Task 10

a)
m = 6
Size = 2^6 = 64 = {0 .. 63}

Succ(n+2i-n)

Server 1
I = 0 : succ(1 + 2^0) = succ(2) = Finger (0) = 5
I = 1 : succ(1 + 2^1) = succ(3) = Finger (1) = 5
I = 2 : succ(1 + 2^2) = succ(5) = Finger (2) = 5
I = 3 : succ(1 + 2^3) = succ(9) = Finger (3) = 32
I = 4 : succ(1 + 2^4) = succ(17) = Finger (4) = 32
I = 5 : succ(1 + 2^5) = succ(33) = Finger (5) = 40

Rest of exercise omitted due to time constraints
Server 60
I = 0 : succ(60 + 2^0-60) = Finger () =
I = 1 : succ(60 + 2^1-60) = Finger () =
I = 2 : succ(60 + 2^2-60) = Finger () =
I = 3 : succ(60 + 2^3-60) = Finger () =
I = 4 : succ(60 + 2^4-60) = Finger () =
I = 5 : succ(60 + 2^5-60) = Finger () =

Server 32
I = 0 : succ(32 + 2^-0) = succ(x) = Finger () =
I = 1 : succ(32 + 2^-1) = succ(x) = Finger () =
I = 2 : succ(32 + 2^-2) = succ(x) = Finger () =
I = 3 : succ(32 + 2^-3) = succ(x) = Finger () =
I = 4 : succ(32 + 2^-4) = succ(x) = Finger () =
I = 5 : succ(32 + 2^-5) = succ(x) = Finger () =

Server 40
I = 0 : succ(40 + 2^-0) = succ(x) = Finger () =
I = 1 : succ(40 + 2^-1) = succ(x) = Finger () =
I = 2 : succ(40 + 2^-2) = succ(x) = Finger () =
I = 3 : succ(40 + 2^-3) = succ(x) = Finger () =
I = 4 : succ(40 + 2^-4) = succ(x) = Finger () =
I = 5 : succ(40 + 2^-5) = succ(x) = Finger () =

Server 5
I = 0 : succ(5 + 2^-1) = succ(x) = Finger () =
I = 1 : succ(5 + 2^-1) = succ(x) = Finger () =
I = 2 : succ(5 + 2^-1) = succ(x) = Finger () =
I = 3 : succ(5 + 2^-1) = succ(x) = Finger () =
I = 4 : succ(5 + 2^-1) = succ(x) = Finger () =

I = 5 : succ(5 + 2^-1) = succ(x) = Finger () =

b)
The server replicas responsible are
5: Server 15
30: Server 32
15: Server 32
60: Server 60
63: Server 1

Using the formula server >= key > pred(server)

c. Using n<k<=succ(n)) to find the immediate successor responsible for k, we first go as far away as we can …

The answer is server replica 40.

d) I) N*(N-1)/2 keys = 1225 keys
   ii) 100 keys. 50 private and 50 public.

e) I would use public/private keys as there is much less keys to manage and each server.