Feedback on Quiz # 3 (Summative)

The Quiz # 3 was composed of 10 questions, which were randomly selected from a Question Bank of 30 questions. Answers and feedback comments for all of the questions are given below:

Q1.1	Which of the following statements is/are true?
(a)	Threads within a process share the same data section.
(b)	The execution of threads within a process can be interleaved.
(c)	Context switching between threads within a process is faster than switching between processes.
(d)	The execution of one thread within a process is guaranteed not to interact with the execution of other threads within that process.
(e)	Each thread maps onto a separate operating system process.
(f)	Each thread within a process maintains its own copy of data.
Feedback: Threads are separate paths of execution within a process: they share the same data section and their execution can be interleaved. This means that switching between the execution of threads (within a process) is much faster than switching between processes.	

Q1.2	Which of the following statements is/are true?
(a)	Threads within a process can interact with each other through shared data.
(b)	The order of execution of statements in different threads within a process is non-deterministic.
(c)	Switching between threads within a process is faster than switching between processes.
(d)	When a new thread is created, the operating system creates a copy of the process' memory, registers and program counter.
(e)	When a thread is running, its data is protected from access by other threads within the same process.
(f)	Every thread within a process must be mapped to a separate kernel thread.
Feedback: Threads are separate paths of execution within a process: they share the same data section, and their execution can be interleaved in a non-deterministic way. This means that switching between threads (within the same process) will be faster but also that there is no protection of data between threads. Often a thread will be mapped to a kernel thread, but this is not necessary.	

Q1.3	Which of the following statements is/are true?
(a)	Race conditions can occur between threads within a process.
(b)	Each thread within a process can be mapped onto a separate kernel thread.
(c)	Every thread within a process shares access to the same memory space.
(d)	Two threads within a process can only interact through system calls.
(e)	A new thread is created using the fork() system call.
(f)	Context switching between threads within the same process is the same as context switching between processes.
Feedback: Threads are separate paths of execution within a process: they share the same data section, and their execution can be interleaved. This means that race conditions can occur between threads within a process. Threads can be mapped on to different kernel threads – but need not be. The fork() system call is used to create a new process (not thread). Normally threads within a process communicate through their shared memory – they do not use system calls. Because threads within a process share most of the resources, a context switch between these will be much faster than between processes.	

Q2.1	Select all the following statements that are true:
(a)	In UDP, a full-duplex connection is established before the two processes can exchange messages.
(b)	TCP and UDP are built on top of the network layer.
(c)	The objective of TCP is to deliver most of the data without errors and in the proper order.
(d)	UDP is used when speed is preferred over error correction and data integrity.
(e)	Neither TCP nor UDP provide any security services.
Feedback: Both TCP and UDP are transport protocols, built on top of the network layer. In TCP, a full-duplex connection is established in which the two processes can send messages to each other over the connection at the same time. Relying on TCP, the communicating processes can deliver all of the data without error and in the proper order. UDP is simpler and faster than TCP, since it is connectionless, so there is no handshaking before the two processes start to communicate. Also, it provides an unreliable data transfer service. Neither TCP nor UDP provide any encryption. TCP-enhanced-with-Secure Sockets Layer provides traditional TCP services in addition to critical process-to-process security services, including encryption, data integrity, and end-point authentication.	

Q2.2	Select all the following statements that are true:
(a)	In TCP, a full-duplex connection is established before the two processes can exchange messages.
(b)	TCP and UDP are built on top of the transport layer.
(c)	The objective of TCP is to deliver all of the data without errors and in the proper order.
(d)	TCP is used when speed is preferred over error correction and data integrity.
(e)	UDP does not include a congestion-control mechanism.

Feedback:

Both TCP and UDP are transport protocols, built on top of the network layer.

In TCP, a full-duplex connection is established in which the two processes can send messages to each other over the connection at the same time.

Relying on TCP, the communicating processes can deliver all of data without error and in the proper order.

UDP is simpler and faster than TCP, since it is connectionless, so there is no handshaking before the two processes start to communicate. Also, it provides an unreliable data transfer service. UDP does not include a congestion-control mechanism, so the sending side of UDP can pump data into the layer below (the network layer) at any rate it pleases.

Q2.3	Select all the following statements that are true:
(a)	In both TCP and UDP, a full-duplex connection is established before the two processes can exchange messages.
(b)	TCP and UDP provide services to the application layer protocols in the TCP/IP stack.
(c)	TCP provides a congestion control mechanism as opposed to UDP, which does not care about network congestion.
(d)	UDP provides no guarantee that the message will reach the receiving process.
(e)	In TCP, when the sending process passes data into its socket, the data travels to the destination process as a ciphertext.

Feedback:

Both TCP and UDP are transport protocols, built below application layer protocols (HTTP and SMTP) in the TCP/IP stack.

In TCP, a full-duplex connection is established in which the two processes can send messages to each other over the connection at the same time.

UDP provides an unreliable data transfer service—that is, when a process sends a message into a UDP

socket, UDP provides no guarantee that the message will ever reach the receiving process. Unlike TCP, UDP does not provide any congestion control mechanism, so the sending side of UDP can pump data into the layer below (the network layer) at any rate it pleases. Neither TCP nor UDP provide any encryption- the data that the sending process passes into its socket is the same data that travels over the network to the destination process.

Q3.1	Circuit switching and packet switching are two different ways of sharing links in a communication network. Which one(s) of the following statement(s) is/are correct?
(a)	Circuit-switched networks require dedicated point-to-point connections during communication.
(b)	In circuit-switched networks, if a circuit segment is idle, its capacity can be used by other connections.
(c)	There is no resource reservation in packet-switched networks, because bandwidth is shared among nodes.
(d)	There is less wastage of bandwidth resources in packet-switched networks as compared to circuit-switched networks.
(e)	Store and forward transmission is supported by circuit-switched networks.
Feedback: Packet-switched networks move data in separate, small blocks known as packets based on the destination address in each packet. When received, packets are reassembled in the proper sequence to make up the message. With packet switching each packet is routed independently across a shared network and the same channel can be used by many nodes to transfer their packets. In circuit switching a connection is established and then data passed over that connection with a guaranteed level of performance. Therefore, the connection is more consistent and the delay between data units is uniform. At the end, the connection is closed. When a circuit segment is idle, it cannot be used by another connection which leads to more wastage of resources. Bandwidth is fixed in this type of switching.	

Q3.2	Circuit switching and packet switching are two different ways of sharing links in a communication network. Which one(s) of the following statement(s) is/are correct?
(a)	Packet-switched networks require dedicated point-to-point connections during communication.
(b)	In circuit-switched networks, if a circuit segment is idle, its capacity cannot be used by other connections.
(c)	In packet-switched networks, resource reservation is possible as the bandwidth is not shared among nodes.
(d)	There is more wastage of bandwidth resources in circuit-switched networks as compared to

	packet-switched networks.
(e)	Store and forward transmission is supported by packet-switched networks.

Feedback:
Packet-switched networks move data in separate, small blocks known as packets based on the destination address in each packet. When received, packets are reassembled in the proper sequence to make up the message. With packet switching each packet is routed independently across a shared network and the same channel can be used by many nodes to transfer their packets.

In circuit switching a connection is established and then data passed over that connection with a guaranteed level of performance. Therefore, the connection is more consistent and the delay between data units is uniform. At the end, the connection is closed. When a circuit segment is idle, it cannot be used by another connection which leads to more wastage of resources. Bandwidth is fixed in this type of switching.

Q3.3	Circuit switching and packet switching are two different ways of sharing links in a communication network. Which one(s) of the following statement(s) is/are correct?
(a)	In circuit-switching, the communication channel from source to destination remains dedicated throughout the data transmission.
(b)	The time delay between data packets in a packet-switched network remains uniform.
(c)	The circuit-switching networks provide a guaranteed quality of service.
(d)	In packet-switching networks, the bandwidth used by nodes remains fixed throughout the communication duration.
(e)	In packet-switching, multiple nodes can use the same channel while transferring their data.

Feedback:
Packet-switched networks move data in separate, small blocks known as packets based on the destination address in each packet. When received, packets are reassembled in the proper sequence to make up the message. With packet switching each packet is routed independently across a shared network and the same channel can be used by many nodes to transfer their packets.

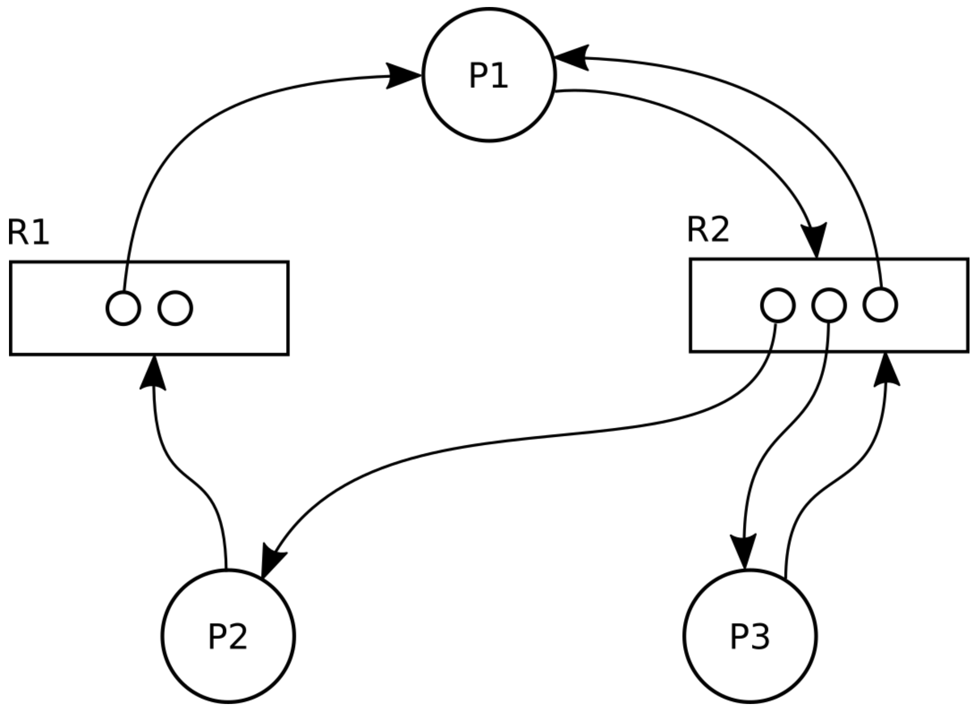
In circuit switching a connection is established and then data passed over that connection with a guaranteed level of performance. Therefore, the connection is more consistent and the delay between data units is uniform. At the end, the connection is closed. When a circuit segment is idle, it cannot be used by another connection which leads to more wastage of resources. Bandwidth is fixed in this type of switching.

Q4.1	Consider the RSA encryption algorithm with p = 3 and q = 7. Which of the following keys are viable RSA public keys K_B^+?
(a)	K_B^+ (n, e) = (21, 11)
(b)	K_B^+ (n, e) = (21, 7)
(c)	K_B^+ (n, e) = (21, 5)
(d)	K_B^+ (n, e) = (21, 3)
(e)	K_B^+ (n, e) = (21, 9)
(f)	K_B^+ (n, e) = (21, 6)
Feedback: Compute n = p x q = 3 x 7 = 21 Compute z = (p-1) x (q-1) = 2 x 6 = 12 Choose e (e < n), such that e is relatively prime with z (has no common factors). So, the viable RSA public keys K_B^+ (n, e) are (21, 5), (21, 7), (21, 11).	

Q4.2	Consider the RSA encryption algorithm with p = 3 and q = 11. Which of the following keys are viable RSA public keys K_B^+?
(a)	K_B^+ (n, e) = (33, 11)
(b)	K_B^+ (n, e) = (33, 9)
(c)	K_B^+ (n, e) = (33, 7)
(d)	K_B^+ (n, e) = (33, 5)
(e)	K_B^+ (n, e) = (33, 14)
(f)	K_B^+ (n, e) = (33, 8)
Feedback: Compute n = p x q = 3 x 11 = 33 Compute z = (p-1) x (q-1) = 2 x 10 = 20 Choose e (e < n), such that e is relatively prime with z (has no common factors). So, the viable RSA public keys K_B^+ (n, e) are (33, 7), (33, 9), (33, 11).	

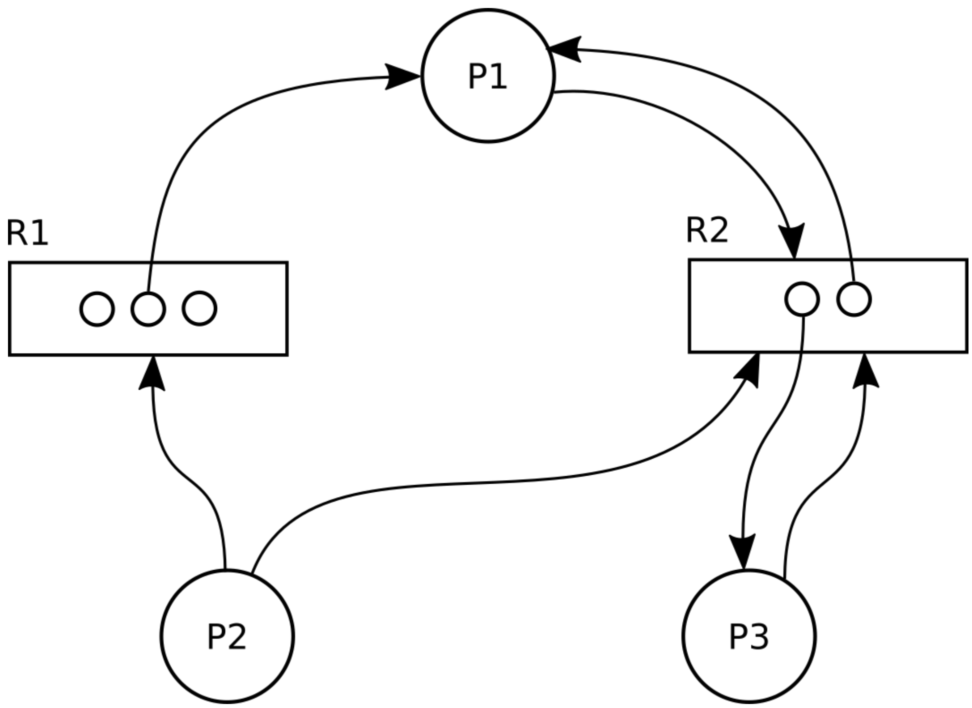
Q4.3	Consider the RSA encryption algorithm with p = 5 and q = 7. Which of the following keys are viable RSA public keys K_B^+?
(a)	K_B^+ (n, e) = (35, 7)
(b)	K_B^+ (n, e) = (35, 19)
(c)	K_B^+ (n, e) = (35, 23)
(d)	K_B^+ (n, e) = (35, 9)
(e)	K_B^+ (n, e) = (35, 21)
(f)	K_B^+ (n, e) = (35, 15)

Feedback:
Compute n = p x q = 5 x 7 = 35
Compute z = (p-1) x (q-1) = 4 x 6 = 24
Choose e (e < n), such that e is relatively prime with z (has no common factors).
So, the viable RSA public keys K_B^+ (n, e) are (35, 7), (35, 19), (35, 23).

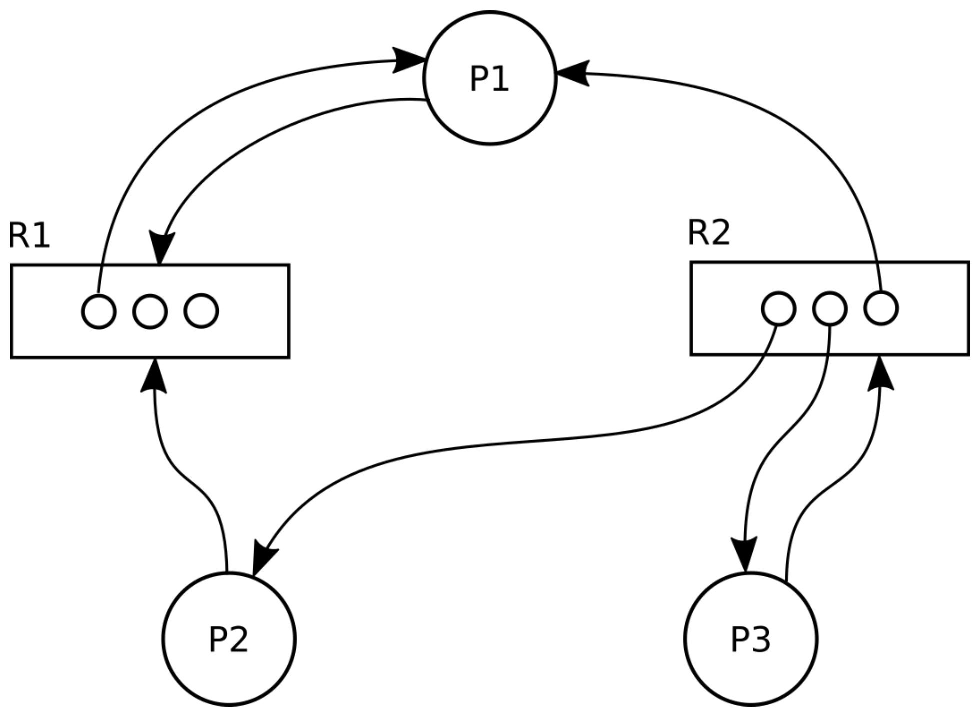
Q5.1	Consider the following resource allocation graph of three processes and two types of resources:  From the following statements, select all that are valid for the above set of processes and resources.
(a)	Processes P1 and P3 are currently blocked.
(b)	Processes P1 and P2 are currently blocked.
(c)	Processes P2 and P3 are currently blocked.
(d)	Process P2 can complete its execution, followed by either P1 or P3.
(e)	Process P3 can complete its execution, followed by either P1 or P2.

Feedback:
In the given RAG:
P1 is holding one instance of R1 and one of R2. It is also requesting one more instance of R2, which is currently not available, hence P1 cannot complete.
P2 is holding an instance of R2 and requesting one instance of R1, which is currently available, therefore, P2 can complete its execution.
P3 is holding an instance of R2 and requesting one more instance of R2, which is currently not

available, hence P3 cannot complete. From the above, we can see that Processes P1 and P3 are blocked. The process P2 can complete its execution, followed by either P1 or P3. The overall system is not deadlocked.

Q5.2	Consider the following resource allocation graph of three processes and two types of resources:  From the following statements, select all that are valid for the above set of processes and resources.
(a)	Processes P1 and P3 are currently blocked but P2 is not blocked.
(b)	Processes P1 and P2 are currently blocked but P3 is not blocked.
(c)	Processes P1, P2 and P3 are currently deadlocked.
(d)	Process P2 can complete its execution, followed by either P1 or P3.
(e)	If we add one more instance to R2, the deadlock will be resolved.
Feedback: In the given RAG: P1 is holding one instance of R1 and one of R2. It is also requesting one more instance of R2, which is currently not available, hence P1 cannot complete. P2 is requesting an instance of R1, which is possible and requesting one instance of R2, which is currently not available, therefore, P2 cannot complete its execution. P3 is holding an instance of R2 and requesting one more instance of R2, which is currently not	

available, hence P3 cannot complete.
From the above, we can see that all processes P1, P2 and P3 are deadlocked. The overall system is currently deadlocked.
If we add one more instance to R2, then process P1 or P2 or P3 may be allocated that instance, therefore, it will be able complete its execution, consequently breaking the circular wait and resolving the deadlock.

<table>
<tr><td>Q5.3</td><td>Consider the following resource allocation graph of three processes and two types of resources:


From the following statements, select all that are valid for the above set of processes and resources.</td></tr>
<tr><td>(a)</td><td>Processes P1 and P2 are currently blocked but P3 is not blocked.</td></tr>
<tr><td>(b)</td><td>Processes P3 is blocked but P1 and P2 are not blocked.</td></tr>
<tr><td>(c)</td><td>Processes P1, P2 and P3 are currently deadlocked.</td></tr>
<tr><td>(d)</td><td>Process P1 can complete its execution, followed by either P2 or P3.</td></tr>
<tr><td>(e)</td><td>Process P3 can complete its execution, followed by either P1 or P2.</td></tr>
<tr><td colspan="2">Feedback:
In the given RAG:
P1 is holding one instance of R1 and one of R2. It is also requesting one more instance of R1, which is available, hence P1 can complete its execution.</td></tr>
</table>

P2 is requesting an instance of R1, which is possible and already holding one instance of R2, therefore, P2 can complete its execution. P3 is holding an instance of R2 and requesting one more instance of R2, which is currently not available, hence P3 cannot complete. From the above, we can see that processes P1 and P2 can complete and only P3 is blocked. The process P1 can complete its execution, followed by either P2 or P3. The overall system is not deadlocked.

Q6.1	A packet propagates over an optical fiber link of distance 3300 km with a transmission rate of 45 Mbps. The length of the packet is 1500 bytes and the propagation speed is 2.5 x 10^8 m/s. Select all the following statements that are true:
(a)	The transmission delay d_{trans} = 266.67 µs
(b)	The propagation delay d_{prop} = 13.20 ms
(c)	The transmission delay d_{trans} = 33.33 µs
(d)	The propagation delay d_{prop} = 13.2 sec
(e)	The propagation delay depends on transmission rate.
(f)	The transmission delay depends on packet length.
Feedback: Propagation delay d_{prop} depends on the length of the physical link (d) and on the propagation speed in medium (s). d = 3300 km s = 2.5 x 10^8 m/s d_{prop} = d/s = 3300 km/ (2.5 x 10^8 m/s) = 13.20 ms. Transmission delay d_{trans} depends on the packet length (L) and the link bandwidth/transmission rate (R). L = 1500 bytes R = 45 Mbps d_{trans} = L/R = 1500 bytes/ 45 Mbps = 1500 x 8 bits / 45 x 10^6 bps = 266.67 µs.	

Q6.2	A packet propagates over an optical fiber link of distance 4500 km with a transmission rate of 13 Mbps. The length of the packet is 2500 bytes and the propagation speed is 2.5 x 10^8 m/s. Select all the following statements that are true:
(a)	The transmission delay d_{trans} = 1.53 ms
(b)	The propagation delay d_{prop} = 18 ms
(c)	The transmission delay d_{trans} = 192.31 µs
(d)	The propagation delay d_{prop} = 18 sec
(e)	The propagation delay depends on the length of the link.

(f)	The transmission delay depends on the propagation speed.
Feedback: Propagation delay d_{prop} depends on the length of the physical link (d) and on the propagation speed in medium (s). d = 4500 km s = 2.5 x 10^8 m/s d_{prop} = d/s = 4500 km / (2.5 x 10^8 m/s) = 18 ms. Transmission delay d_{trans} depends on the packet length (L) and the link bandwidth/transmission rate (R). L = 2500 bytes R = 13 Mbps d_{trans} = L/R = 2500 bytes / 13 Mbps = 2500 x 8 bits / 13 x 10^6 bps = 1.53 µs.	

Q6.3	A packet propagates over an optical fiber link of distance 1700 km with a transmission rate of 5 Mbps. The length of the packet is 1950 bytes and the propagation speed is 2.5 x 10^8 m/s. Select all the following statements that are true:
(a)	The transmission delay d_{trans} = 3.12 ms
(b)	The propagation delay d_{prop} = 6.8 ms
(c)	The transmission delay d_{trans} = 0.39 ms
(d)	The propagation delay d_{prop} = 6.8 sec
(e)	The propagation delay depends on the packet length.
(f)	The transmission delay depends on the bandwidth.
Feedback: Propagation delay d_{prop} depends on the length of the physical link (d) and on the propagation speed in medium (s). d = 1700 km s = 2.5 x 10^8 m/s d_{prop} = d/s = 1700 km/ (2.5 x 10^8 m/s) = 6.8 ms. Transmission delay d_{trans} depends on the packet length (L) and the link bandwidth/transmission rate (R). L = 1950 bytes R = 5 Mbps d_{trans} = L/R = 1950 bytes/ 5 Mbps = 1950 x 8 bits / 5 x 10^6 bps = 3.12 ms.	

Q7.1	An IPv4 fragment has arrived with an offset value of 145 and a length of 1000 bytes. The numbers of the first and last bytes are: **Note:** The number refers to the position in the overall datagram, after the fragments are reassembled.
(a)	The first byte number is 1160 and the last byte number is 2139.
(b)	The first byte number is 1160 and the last byte number is 2159.
(c)	The first byte number is 145 and the last byte number is 1124.
(d)	The first byte number is 145 and the last byte number is 1144.
Feedback: The fragmentation offset field is a 13-bit field that shows the relative position of this fragment with respect to the whole datagram, measured in units of 8 bytes. To find the number of the first byte, we multiply the offset value by 8. This means that the first byte number is 1160. We can determine the last byte number by subtracting 20 from the length (for the header field) and then adding it to the first byte number minus one i.e. 1160 + (1000 – 20) – 1 = 2139	

Q7.2	An IPv4 fragment has arrived with an offset value of 225 and a length of 1000 bytes. The numbers of the first and last bytes are: **Note:** The number refers to the position in the overall datagram, after the fragments are reassembled.
(a)	The first byte number is 1800 and the last byte number is 2779.
(b)	The first byte number is 1800 and the last byte number is 2799.
(c)	The first byte number is 225 and the last byte number is 1204.
(d)	The first byte number is 225 and the last byte number is 1224.
Feedback: The fragmentation offset field is a 13-bit field that shows the relative position of this fragment with respect to the whole datagram, measured in units of 8 bytes. To find the number of the first byte, we multiply the offset value by 8. This means that the first byte number is 1800. We can determine the last byte number by subtracting 20 from the length (for the header field) and then adding it to the first byte number minus one i.e. 1800 + (1000 – 20) – 1 = 2779	

Q7.3	An IPv4 fragment has arrived with an offset value of 515 and a length of 1000 bytes. The numbers of the first and last bytes are: **Note:** The number refers to the position in the overall datagram, after the fragments are reassembled.
(a)	The first byte number is 4120 and the last byte number is 5099.
(b)	The first byte number is 4120 and the last byte number is 5119.
(c)	The first byte number is 515 and the last byte number is 1494.
(d)	The first byte number is 515 and the last byte number is 1514.

Feedback:
The fragmentation offset field is a 13-bit field that shows the relative position of this fragment with respect to the whole datagram, measured in units of 8 bytes.
To find the number of the first byte, we multiply the offset value by 8. This means that the first byte number is 4120. We can determine the last byte number by subtracting 20 from the length (for the header field) and then adding it to the first byte number minus one i.e. 4120 + (1000 – 20) – 1 = 5099

Q8.1	A resource allocation system that uses Banker's algorithm for 3 resource types (A, B, C) and 4 processes (P0, P1, P2, P3) is currently in the following state. (Claim Matrix: max need of each process. Allocation Matrix: resources held by each process. Need Matrix: resources needed to complete. Available: free resources.)

Claim Matrix

	A	B	C
P0	4	3	3
P1	3	2	2
P2	9	0	2
P3	1	1	2

Allocation Matrix

	A	B	C
P0	1	1	2
P1	2	1	2
P2	4	0	1
P3	1	1	2

Need Matrix

	A	B	C
P0	3	2	1
P1	1	1	0
P2	5	0	1
P3	0	0	0

Available Resource Vector

A	B	C
2	1	0

Presuming that the system is currently in Safe State, which one(s) of the following safe sequence(s) is/are possible?

(a)	P1, P3, P0, P2
(b)	P1, P3, P2, P0
(c)	P1, P2, P0, P3
(d)	P1, P2, P3, P0
(e)	The presumption is incorrect, the system is not in a Safe State.

Feedback:

Either P1 or P3 can run to completion but since the options start with P1 we will take it. So, we will have the following available vector:

A	**B**	**C**
4	2	2

Either P0 or P3 can run to completion but since the options continued with P3 we will take it. So, we will have the following available vector:

A	**B**	**C**
5	3	4

Either P0 or P2 can run to completion, let's take P0. So, we will have the following available vector:

A	**B**	**C**
6	4	6

Let's complete P2 now:

A	**B**	**C**
10	4	7

So, the correct options are:

P1, P3, P0, P2

P1, P3, P2, P0

Q8.2 A resource allocation system that uses Banker's algorithm for 3 resource types (A, B, C) and 4 processes (P0, P1, P2, P3) is currently in the following state.
(Claim Matrix: max need of each process. Allocation Matrix: resources held by each process. Need Matrix: resources needed to complete. Available: free resources.)

Claim Matrix

	A	B	C
P0	4	3	3
P1	3	3	2
P2	1	1	2
P3	9	0	2

Allocation Matrix

	A	B	C
P0	1	1	2
P1	2	1	2
P2	1	1	2
P3	4	0	1

Need Matrix

	A	B	C
P0	3	2	1
P1	1	2	0
P2	0	0	0
P3	5	0	1

Available Resource Vector

A	B	C
2	1	0

	Presuming that the system is currently in Safe State, which one(s) of the following safe sequence(s) is/are possible?
(a)	P2, P1, P0, P3
(b)	P2, P1, P3, P0
(c)	P2, P3, P0, P1
(d)	P2, P3, P1, P0
(e)	The presumption is incorrect, the system is not in a Safe State.

Feedback:

Only P2 can run to completion. So, we will have the following available vector:

A	B	C
3	2	2

Either P0 or P1 can run to completion but since the options continued with P1 we will take it. So, we will have the following available vector:

A	B	C
5	3	4

Either P0 or P3 can run to completion, let's take P0. So, we will have the following available vector:

A	B	C
6	4	6

Let's complete P3 now:

A	B	C
10	4	7

So, the correct options are:

P2, P1, P0, P3

P2, P1, P3, P0

Q8.3

A resource allocation system that uses Banker's algorithm for 3 resource types (A, B, C) and 4 processes (P0, P1, P2, P3) is currently in the following state.
(Claim Matrix: max need of each process. Allocation Matrix: resources held by each process. Need Matrix: resources needed to complete. Available: free resources.)

Claim Matrix

	A	B	C
P0	6	3	3
P1	3	3	2
P2	2	2	2
P3	8	0	2

Allocation Matrix

	A	B	C
P0	1	1	2
P1	2	1	2
P2	2	1	2
P3	4	0	1

Need Matrix

	A	B	C
P0	5	2	1
P1	1	2	0
P2	0	1	0
P3	4	0	1

Available Resource Vector

A	B	C
2	1	0

Presuming that the system is currently in Safe State, which one(s) of the following safe sequence(s) is/are possible?

(a)	P2, P3, P0, P1
(b)	P2, P3, P1, P0
(c)	P2, P0, P3, P1
(d)	P2, P0, P1, P3
(e)	The presumption is incorrect, the system is not in a Safe State.

Feedback:

Only P2 can run to completion. So, we will have the following available vector:

A	**B**	**C**
4	2	2

Either P3 or P1 can run to completion but since the options continued with P3 we will take it. So, we will have the following available vector:

A	**B**	**C**
8	2	3

Either P0 or P1 can run to completion, let's take P0. So, we will have the following available vector:

A	**B**	**C**
9	3	5

Let's complete P1 now:

A	**B**	**C**
11	4	7

So, the correct options are:
P2, P3, P0, P1
P2, P3, P1, P0

Q9.1	We know that RDT3.0 is correct, but it suffers greatly in terms of performance due to being a Stop-and-Wait protocol. Let's assume that we have a 2 Gbps link between two devices and 40ms propagation delay, what will be the throughput on this link for a packet size of 4 KBytes when using RDT3.0 protocol?
(a)	**399.92 Kb/sec**
(b)	**49.99 KB/sec**
(c)	49.99 Kb/sec
(d)	6.25 KB/sec
(e)	399.92 KB/sec

Feedback:
We will compute the transmission delay, before we can compute the utilization and throughput:
$d_{trans} = L/R = (4000 * 8) / (2 \times 10^9) = 16$ microseconds
$U_{sender} = (L/R) / (RTT + L/R) = 0.016 / (80 + 0.016) = 0.00019996$
Throughput = $2 \times 10^9 \times 0.00019996 = 399.92$ Kbps = 49.99 KB/sec

Q9.2	We know that RDT3.0 is correct, but it suffers greatly in terms of performance due to being a Stop-and-Wait protocol. Let's assume that we have a 10 Gbps link between two devices and 50ms propagation delay, what will be the throughput on this link for a packet size of 500 Bytes when using RDT3.0 protocol?
(a)	**39.99 Kb/sec**
(b)	**4.99 KB/sec**
(c)	4.99 Kb/sec
(d)	0.625 KB/sec
(e)	39.99 KB/sec

Feedback:
We will compute the transmission delay, before we can compute the utilization and throughput:
$d_{trans} = L/R = (500 * 8) / (10 \times 10^9) = 0.4$ microseconds
$U_{sender} = (L/R) / (RTT + L/R) = 0.0004 / (100 + 0.0004) = 0.000004$
Throughput = $10 \times 10^9 \times 0.000004 = 39.99$ Kbps = 4.99 KB/sec

Q9.3	We know that RDT3.0 is correct, but it suffers greatly in terms of performance due to being a Stop-and-Wait protocol. Let's assume that we have a 3 Gbps link between two devices and 60ms propagation delay, what will be the throughput on this link for a packet size of 2 KBytes when using RDT3.0 protocol?
(a)	**133.33 Kb/sec**
(b)	**16.67 KB/sec**
(c)	16.67 Kb/sec
(d)	2.08 KB/sec
(e)	133.33 KB/sec

Feedback:

We will compute the transmission delay, before we can compute the utilization and throughput:

d_{trans} = L/R = (2000 * 8) / (3 x 10^9) = 5.333333 microseconds

U_{sender} = (L/R) / (RTT + L/R) = 0.005333333 / (120 + 0.005333333) = 0.000044442

Throughput = 3 x 10^9 x 0.000044442 = 133.33 Kbps = 16.67 KB/sec

Q10.1	Which of the following statement(s) is/are true?
(a)	A DNS server is used to resolve a domain name to an IP address
(b)	TCP ensures that all data arrives correctly and in order
(c)	The HTTPS protocol is part of the transport layer
(d)	A firewall guarantees that your network is secure
(e)	A router is functionally equivalent to a switch

Feedback:

A DNS server is used to map between a domain name (e.g. www.bham.ac.uk) and its IP address.

TCP will ensure a reliable connection. So it will, for instance, ensure that the data is assembled in the correct order and that all data is received.

HTTPS is a protocol used to establish a secure connection. It is not part of the transport layer.

Whilst a firewall will greatly improve the security of a network it cannot protect against all attacks.

A router works at the IP level whereas a switch is at the network level

<table>
<tr><td>Q10.2</td><td>Which of the following statement(s) is/are true?</td></tr>
<tr><td>(a)</td><td>A firewall can be used to restrict external access to computers and services within an intranet</td></tr>
<tr><td>(b)</td><td>UDP does not guarantee that packets will arrive in order</td></tr>
<tr><td>(c)</td><td>In TCP, when a connection has been established, this defines the route for all packets until the connection is closed</td></tr>
<tr><td>(d)</td><td>SMTP is the protocol used for resolving a domain name to an IP address</td></tr>
<tr><td>(e)</td><td>HTTP uses UDP for the connection between the client and server</td></tr>
<tr><td colspan="2">Feedback:
A firewall is typically used to protect an intranet. It will do this in several ways one of which is by restricting which computers and services within the intranet can be accessed from outside.
If you need a reliable connection then you should use TCP. UDP is lighter weight but does not guarantee the correct order of packets.
With TCP a connection is established but packets may be routed differently.
SMTP (Simple Mail Transfer Protocol) is used for delivery of mail. DNS is the service to resolve domain names to IP addresses.
HTTP requires a reliable service and is built on TCP.</td></tr>
</table>

<table>
<tr><td>Q10.3</td><td>Which of the following statement(s) is/are true?</td></tr>
<tr><td>(a)</td><td>TCP provides a connection-based protocol that provides a reliable service</td></tr>
<tr><td>(b)</td><td>SMTP is a protocol used to transmit email</td></tr>
<tr><td>(c)</td><td>A DNS server is used to resolve a domain address to an IP address</td></tr>
<tr><td>(d)</td><td>With UDP it is guaranteed that all packets will arrive in the correct order</td></tr>
<tr><td>(e)</td><td>With TCP the route between a client and server is fixed when a connection is established</td></tr>
<tr><td colspan="2">Feedback:
TCP establishes a connection between machines. It delivers a reliable service that ensures that data is error free and correctly ordered.
SMTP is the most common protocol for delivery of email
DNS is a service that is used for determining the mapping between a domain address (e.g. bham.ac.uk) and its IP address.
UDP does not guarantee that data is delivered reliably or in order. This is acceptable as a trade off in services like audio streaming.
Although TCP establishes a connection it is not necessary that all packets follow the same route.</td></tr>
</table>