Started on	Wednesday, 4 August 2021, 11:39 AM
State	Finished
Completed on	Thursday, 5 August 2021, 4:56 PM
Time taken	1 day 5 hours
Grade	**10.00** out of 10.00 (**100**%)

Information

We consider **circuit switching**.

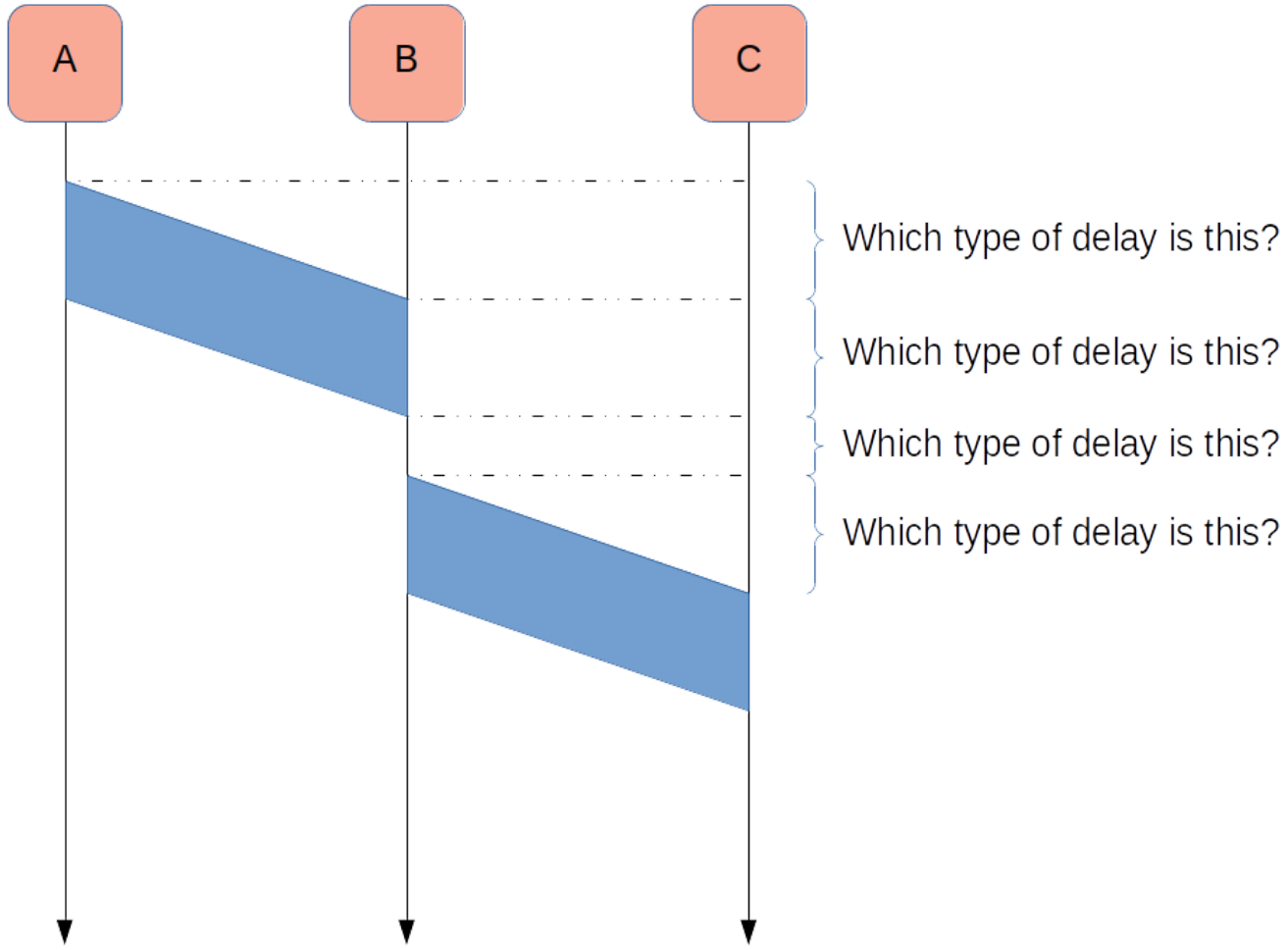
We are given a system with three stations A, B, C such that A is connected to B, and B is connected to C. A wants to establish a circuit to C, which has to go through B.

We are interested in the time it takes to go through the "unproductive" connection setup phase, which we have to complete before we can transmit any data. The length of the links A - B and B - C is 10,000 km each, the speed of light on the cables is 200,000 km/s. The data rate supported on both links is 1 Mbps, and there are no transmission errors on the links.

To establish a circuit, station A will send a particular message, the **call-setup-request** message, of 1,000 bits length to B. After receiving this message, station B will need a time of 1 ms to process it, before B continues to send the same message further on to C. Once C has fully received the message, it will process it (which again takes 1 ms) and then generate a **call-setup-response** message of 1,000 bits length, which it sends back to A (through B). After B has fully received the call-setup-response message, it will process it (taking 1 ms) and forward it to A. After A has completely received it, A will process it (which takes 1 ms) and after that station A can commence with actual data transmission.

In the following, all times to be calculated are to be given in seconds, unless explicitly said otherwise.

It will be helpful to draw a fairly detailed time-event diagram, e.g. in the way shown below (where time flows from the upper to the lower part of the picture). Work out what the different types of delays are (e.g. propagation delay, processing delay, transmission delay).



Question **1**

Correct

Mark 1.00 out of 1.00

Let *L* stand for the length of one link in km, *c* stand for the speed of light on the cable (in km/s), *R* stand for the data rate available on either of the links (in bps), *M* stand for the length of the **call-setup-*** messages (in bits), and *P* stand for the processing times required by A, B and C (in seconds). Find a general expression for the duration of the connection setup phase (i.e. the time between A starting the process and A being able to commence data transmission) and implement it as a Python function.

For example:

Test	Result
`print ("{:.4f}".format(connection_setup_delay(10000, 200000, 1000000, 1000, 0.001)))`	`0.2080`

Answer: (penalty regime: 10, 20, ... %)

Reset answer

```
def connection_setup_delay (cableLength_km, speedOfLight_kms, dataRate_bps, messageLength_b, processingTimes_s):

    p_delay = cableLength_km/speedOfLight_kms
    t_delay = messageLength_b/dataRate_bps

    return (p_delay + t_delay + processingTimes_s) * 4
```

	Test	Expected	Got	
✔	`print ("{:.4f}".format(connection_setup_delay(10000, 200000, 1000000, 1000, 0.001)))`	`0.2080`	`0.2080`	✔
✔	`print ("{:.4f}".format(connection_setup_delay(20000, 250000, 2000000, 800, 0.002)))`	`0.3296`	`0.3296`	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

Question **2**

Correct

Mark 1.00 out of 1.00

Evaluate the expression you have developed in the previous question for the following values: link length *L* = 10,000 km, speed of light *c* = 200,000 km/s, data rate *R* = 1 Mbps, message length *M* = 4,000 bits, and processing time P = 0.001 s. Give the value in seconds to two digits after the decimal point, no rounding.

Answer: 0.22 ✔

Correct

Marks for this submission: 1.00/1.00.

Question **3**

Correct

Mark 1.00 out of 1.00

Once the call has been established (i.e. we have completed the connection setup phase), any data signal sent by A will be forwarded by B without any additional delay, i.e. during the data forwarding phase (or connection usage phase) things appear as if B is not present at all. If we call the duration of the connection setup phase *TS*, and we want to transmit a message of length *M* bits, find a general expression for the total time that passes between A starting the connection setup and C receiving the last bit of the message, assuming that A commences with data transfer immediately after it has received and processed the **call-setup-response message**. We call this the **message delay**. Please implement your general expression as a Python function.

For example:

Test	Result
`print ("{:.3f}".format(message_delay(0.305, 15000, 200000, 5000, 1000000)))`	`0.460`

Answer: (penalty regime: 10, 20, ... %)

Reset answer

```
def message_delay (connSetupTime_s, cableLength_km, speedOfLight_kms, messageLength_b, dataRate_bps):

    p_delay = cableLength_km/speedOfLight_kms
    t_delay = messageLength_b/dataRate_bps

    return connSetupTime_s + ((p_delay) * 2) + t_delay
```

	Test	Expected	Got	
✔	`print ("{:.3f}".format(message_delay(0.305, 15000, 200000, 5000, 1000000)))`	`0.460`	`0.460`	✔
✔	`print ("{:.3f}".format(message_delay(0.208, 20000, 200000, 10000, 1000000)))`	`0.418`	`0.418`	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

Question **4**

Correct

Mark 1.00 out of 1.00

Assume that the connection setup delay is 0.2 seconds. Assume furthermore the following parameters: link length *L* = 10,000 km, speed of light *c* = 200,000 km/s and data rate *R* = 1 Mbps. We want to transmit a message of *M* = 1,000 bits size. What is the message delay in seconds? Please give three digits after the decimal point, no rounding.

Answer: 0.301 ✔

Correct

Marks for this submission: 1.00/1.00.

Question **5**

Correct

Mark 1.00 out of 1.00

Same setup as previous question, but now we want to transmit a message of M = 1,000,000,000 bits size. What is the message delay in seconds? Please give one digit after the decimal point, no rounding.

Answer: 1000.3

Correct

Marks for this submission: 1.00/1.00.

Information

We now consider the case of **packet switching**.

As before, we are given a system with three stations A, B, C such that A is connected to B and B is connected to C. A wants to transmit data to C and has to go through B for that.

In packet switching, a large message is split up into packets. Each packet can carry a maximum number S of user data bits and will also have a number O of overhead bits (which for example contain address fields for the sender and receiver of the packet, or a packet checksum to allow the receiver to check the correctness). This overhead O will occur for each and every packet, even if it carries less than S user data bits.

A key feature of packet switching is that no connection or call setup or teardown phases are required to send packets.

Again it will be helpful to draw a time-event diagram.

In the following, all times to be calculated are to be given in seconds, unless explicitly said otherwise.

Question **6**

Correct

Mark 1.00 out of 1.00

Suppose a packet can carry at most *S* bits of user data and requires *O* bits of overhead. We are given a message of *M* bits, and *M* is not necessarily an integer multiple of *S*. Please find an expression for the total number of bits (user data and overhead bits) that need to be transmitted to transmit all *M* user data bits in packets. Please implement your expression as a Python function.

The math Python library provides functions like math.floor or math.ceil that can be used for rounding.

For example:

Test	Result
`print ("{:.1f}".format(total_number_bits(1000, 100, 10000)))`	11000.0

Answer: (penalty regime: 10, 20, ... %)

Reset answer

```
import math
def total_number_bits (maxUserDataBitsPerPacket_b, overheadBitsPerPacket_b, messageLength_b):
    S = maxUserDataBitsPerPacket_b
    O = overheadBitsPerPacket_b
    M = messageLength_b #not necessarily an integer multiple of S.

    total_packages = math.ceil(M/S)
    total_O = total_packages * O

    return messageLength_b + total_O
```

	Test	Expected	Got	
✔	`print ("{:.1f}".format(total_number_bits(1000, 100, 10000)))`	11000.0	11000.0	✔
✔	`print ("{:.1f}".format(total_number_bits(1000, 100, 10001)))`	11101.0	11101.0	✔
✔	`print ("{:.1f}".format(total_number_bits(1000, 100, 10999)))`	12099.0	12099.0	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

Question **7**

Correct

Mark 1.00 out of 1.00

The intermediate router B can only start to process a packet (e.g. checking its checksum, figuring out the next hop) once it has completely received the packet. The actual processing time of a router is *P* seconds. The receiving node C also needs to spend processing time *P* on the packet. Suppose station A starts to send a packet with *S* user data bits and *O* overhead bits to station C. Find a general expression for the time between A starting the transmission of this packet and C having processed the received packet, depending on *L* (the length of a link in km), *C* (speed of light in km/s), *P* (processing delay in B and C in seconds), *R* (data rate in bps) and *O* and S. We call this time the **packet transfer time**. Please implement your expression as a Python function.

For example:

Test	Result
`print ("{:.4f}".format(packet_transfer_time(10000, 200000, 0.001, 1000000, 1000, 100)))`	`0.1042`

Answer: (penalty regime: 10, 20, ... %)

Reset answer

```
def packet_transfer_time (linkLength_km, speedOfLight_kms, processingDelay_s, dataRate_bps, maxUserDataBitsPerPacke
    L = linkLength_km
    C = speedOfLight_kms
    P = processingDelay_s
    R = dataRate_bps
    S = maxUserDataBitsPerPacket_b
    O = overheadBitsPerPacket_b

    p_delay = L/C
    t_delay = (S+O)/R


    return (p_delay + t_delay + P) * 2
```

	Test	Expected	Got	
✔	`print ("{:.4f}".format(packet_transfer_time(10000, 200000, 0.001, 1000000, 1000, 100)))`	`0.1042`	`0.1042`	✔
✔	`print ("{:.4f}".format(packet_transfer_time(10000, 200000, 0.001, 3000000, 1000, 100)))`	`0.1027`	`0.1027`	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

Question **8**

Correct

Mark 1.00 out of 1.00

Please calculate this packet transfer time for a packet with s = 4,192 bits user data, o = 100 bits overhead and a processing delay of P = 1 ms in nodes B and C. Use L = 15,000 km, c = 250,000 km/s, R = 1 Mbps. Please give the answer in seconds with four digits after the decimal point, no rounding (while working on it, print the answer to at least five digits after the decimal point and take the first four ones).

In other words, evaluate 'packet_transfer_time (15000, 250000, 0.001, 1000000, 4192, 100)'.

Answer: 0.1305 ✔

Correct

Marks for this submission: 1.00/1.00.

Question **9**

Correct

Mark 1.00 out of 1.00

Now suppose that A has a message of size *M* bits which is an integer multiple of the maximum packet user data size of *S* bits. Station A prepares *M/S* packets and sends them back-to-back, without any gap. Nodes B and C can also process incoming packets without a gap: if a node has finished processing one packet (which takes *P* seconds processing time) and the next packet has been completely received at this time, processing of this next packet can start immediately and we have a kind of "pipelining effect". With this, and re-using the general expression for the time by which C will have processed the first packet, find a general expression for the time by which C will have processed all *M/S* packets (use the simplification that *M/S* is an integer and there are hence no slack packets). Implement your expression as a Python function.

Note that station B can process incoming packets and transmit outgoing packets at the same time.

For example:

Test	Result
`print ("{:.4f}".format(total_transfer_time(20000, 200000, 0.001, 1000000, 1000, 100, 5000)))`	`0.2086`

Answer: (penalty regime: 10, 20, ... %)

Reset answer

```
def total_transfer_time (linkLength_km, speedOfLight_kms, processingDelay_s, dataRate_bps, maxUserDataBitsPerPacket
    l = linkLength_km
    c = speedOfLight_kms
    p = processingDelay_s
    r = dataRate_bps
    s = maxUserDataBitsPerPacket_b
    o = overheadBitsPerPacket_b
    m = messageLength_b

    p_delay = l/c
    t_delay = (s+o)/r

    total_packages = (m/s)

    return p_delay + t_delay + p + p_delay + (total_packages * t_delay) + p
```

	Test	Expected	Got	
✔	`print ("{:.4f}".format(total_transfer_time(20000, 200000, 0.001, 1000000, 1000, 100, 5000)))`	`0.2086`	`0.2086`	✔
✔	`print ("{:.4f}".format(total_transfer_time(20000, 200000, 0.001, 1000000, 1000, 100, 50000)))`	`0.2581`	`0.2581`	✔
✔	`print ("{:.4f}".format(total_transfer_time(25000, 200000, 0.001, 1000000, 1000, 100, 5000)))`	`0.2586`	`0.2586`	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

Question **10**

Correct

Mark 1.00 out of 1.00

Evaluate the previously developed expression for the total transfer time for link length *L* = 10,000 km, speed of light *c* = 200,000 km/s, data rate *R* = 1 Mbps, processing time P = 0.001 seconds, maximum user data size per packet *s* = 1,000 bits, overhead per packet *o* = 100 bits, and message size *M* = 1,000,000,000 bits. Please give your answer in seconds with four digits after the decimal point, no rounding.

In other words, evaluate 'total_transfer_time (10000, 200000, 0.001, 1000000, 1000, 100, 1000000000)'.

Answer: 1100.1031 ✔

Correct

Marks for this submission: 1.00/1.00.

◄ Quiz: Networked applications, QoS (Practice copy)

Jump to...

Quiz: Circuit- and packet switching (Practice copy) ►