Assignment 2 FAQs

1. **What is the ideal timeout value?**

   You can use 100 milliseconds. But you can try other values as well.
   Please do not use values greater than 1 second, otherwise testing your program would take too much time.

2. **How much data should be in each packet?**

   Each packet has to have exactly 500 bytes data unless the number of remaining characters is less than 500 (only occurs for the last packet).
   So each packet (other than last data packet and EOT) has to be exactly 512 bytes (4 bytes for type + 4 bytes for seqnum + 4 bytes for data length + 500 bytes of data).

3. **What is the seqnum of the EOT packet?**

   The seqnum of the EOT packet should be equal to ((1 + <seqnum-of the-last-data-packet>) mod 32).

4. **Do input files have any special character that could cause some encoding/decoding problems?**

   No, all the input files contain only alphabetic ([a-z] and [A-Z]) characters.

5. **What are the possible values for the emulator max delay?**

   It could get any value **greater** than 0.

6. **What are the possible values for the emulator packet discard probability?**

   The discard probability has to be >= 0 and < 1. Probability 0 means that the emulator does not drop any packet at all.

7. **I am trying to run the emulator and I get the following error. What does it mean?**

   ```
   Can't bind local address in backward receiving.
   : Address already in use
   ```

   It means that the UDP port number in the backward direction that you use for running the emulator is not open (It is used by some other program).
   To solve this problem you can use the following command. It will return 3 random open ports between 1000 and 2000.

```
comm -23 <(seq 1000 2000 | sort) <(ss -tan | awk '{print $4}' | cut
-d':' -f2 | grep "[0-9]\{1,5\}" | sort -u) | shuf | head -n 3
```

* Same error could happen for UDP forward receiving port as well.
* You can use `ctrl + c` to terminate the emulator, otherwise, you may not be able to use the same port numbers again.

8. **Do I need to implement the packet class if I want to implement in a language other than Java?**

   Yes, you have to implement the packet class yourself. You must use exactly the same encoding approach (as packet.java) for your packet class since the emulator only works with that encoding. You can check your encoding by running the emulator in the verbose mode. If the seqnum that the emulator prints are similar to your packets seqnums then your encoding is fine, otherwise (the seqnums are some very large or negative values) it means that your encoding is not correct.

   *** array() function from Bytebuffer class in java use big-endian order to convert a Bytebuffer instance to a byte array. So you have to use the same endianness to create a byte array from the UDP data.

9. **Should I use multithreading for implementing the assignment?**

   Sender: We strongly suggest you to use multithreading to implement the sender (one thread for sending packets and one for listening for the acks). I am not aware of any kind of (completely correct) implementation without multithreading that satisfies all the assignment requirements. There are some approaches which use only a single thread but you should expect to lose some marks if they could not satisfy all the requirements of the A2.

   Receiver: Receiver could be implemented using a single thread and there is no need of using multithreading.

Useful information:

1. Since we run sender, emulator, and receiver on 3 different machines for marking, make sure that your programs work properly in this setting.
2. You don't need to send a Makefile if you are using a scripting language like python, otherwise, you have to have a Makefile.
3. Sender and receiver have to terminate after receiving EOT packet (receiver has to send an EOT and then exit).