Problem 1

Below is an IP packet captured by Wireshark. Bytes are shown in HEX mode, grouped by 4 HEX digits. We know that this packet is using IPv4 with no option field.

```
__ 00 00 1e
a7 f8 00 00
ff 11 __ __
ac 1f 09 ee
80 61 80 01
d4 2d 00 35
00 0a __ __
61 61
```

1. What should be the HEX value for the first blank (line 1)?
2. What should be the HEX value for the second blank (two HEX digits, line 3)?
3. What should be the HEX value for the third blank (two HEX digits, line 7)?
4. What is the transport layer protocol carried in this IP packet? How do you know that?
5. What is the payload carried in the transport layer protocol you identified?

1. The first should be 0x45, for IPv4 and a header length of $5 \times 4 = 20$ bytes.
2. The second blank is the IP header checksum, which is calculated as follows:
   ```
   0x4500 + 0x001E + // version, IHL, <unused>, total length
   0xA7F8 + 0x0000 + // identification, flags, fragment off
   0xFF11 + 0x0000 + // TTL, protocol, checksum (zero)
   0xAC1F + 0x09EE + // source IP address
   0x8061 + 0x8001 = 0x3A296 // dest IP address
   ```
   With wraparound: `0xA296 + 0x0003 = 0xA299`
   With bit flip: `0x5D66`
3. The third blank is the UDP checksum:
   ```
   0xAC1F + 0x09EE + // source IPv4 address
   0x8061 + 0x8001 + // destination IPv4 address
   0x0011 + 0x000A + // zeroes, UDP protocol, UDP length
   0xD42D + 0x0035 + // source port, destination port
   0x000A + 0x0000 + // length, checksum (zero)
   0x6161 = 0x2EC57 // data
   ```
   With wraparound: `0xEC57 + 0x0002 = 0xEC59`
   With bit flip: `0x13A6`
4. The protocol is UDP, as the protocol field in the IP header reads $11_{16} = 17_{10}$, which is defined as UDP in RFC 790.
5. The payload of this UDP packet is `0x6161`.

Problem 2

Assume that the pipeline sizes for the Go-back-N and TCP (no delayed ACKs) protocols are large enough such that 5 consecutive data segments and their corresponding ACKs can be received (if not lost in the channel) by the receiving host (Host B) and the sending host (Host A) respectively. Suppose Host A sends 5 data segments to Host B, and the 2nd segment (sent from A) is lost. This loss is then detected by the protocol-specific means and the protocol invokes recovery actions. In the end, all 5 data segments are correctly received by Host B.

1. In case of Go-back-N, how many segments has Host A sent in total and how many ACKs has Host B sent in total?
2. In case of TCP (no delayed ACKs), how many segments has Host A sent in total and how many ACKs has Host B sent in total?
3. If the timeout value happen to be set to 10 RTT, which of the protocol successfully delivers all five data segments in shortest time interval and give estimate of this time?

1. In this case, Host A sends 5 segments to Host B but the second segment was lost. Host B sends 4 ACKs in response, all acknowledging the receipt of the first segment. After the timeout expires from not haveing received any ACKs for the segments in its current window, Host A resends segments #2 through #5 at which point Host B has received all 5 segments and ACKS segments #2 through #5.

   The number of segments sent is 5 + 4 = 9 and the number of ACKs sent is 4 + 4 = 8.

2. In this case, Host A sends 5 segments to Host B, but segment #2 is lost. Host B ACKs segment #2 4 times, but buffers the other segments that it received out of order. Host A resends segment #2 and Host B sends an ACK for segment #6 indicating that it has already buffered segments #3 - #5.

   The number of segments sent is 5 + 1 = 6 and the number of ACKs sent is 4 + 1 = 5.

3. If we make the simplifying assumption that we can send 5 packets on the network at the same time, and that TCP fast retransmit is set, TCP will take the least amount of time to deliver all 5 segments:

   1RTT (send first 5 segments simultaneously and ACK segment #2 4 times, triggering fast retransmit)
   +
   1RTT (send the missing packet #2, it arrives in half a round trip and then Host B must acknowledge)
   = 2RTT

   If we assume Go-back-N, we have to wait 1RTT for the first volley of packets, then we have to wait for the timeout to expire, and then we have to wait another RTT to resend the remaining 4 packets, for a grand total of 12 RTTs.

Problem 3

Host A and B are communicating over a TCP connection, and Host B has already received from A all bytes up through byte 126. Suppose Host A then sends two segments to Host B back-to-back. The first and second segments contain 80 and 40 bytes of data, respectively. In the first segment, the sequence number is 127, the source port number is 302, and the destination port number is 80. Host B sends an acknowledgment whenever it receives a segment from Host A.

1. In the second segment sent from Host A to B, what are the sequence number, source port number, and destination port number?
2. If the first segment arrives before the second segment, in the acknowledgment of the first arriving segment, what is the acknowledgment number, the source port number, and the destination port number?
3. If the second segment arrives before the first segment, in the acknowledgment of the first arriving segment, what is the acknowledgment number?

1. sequence number: 207
   source port: 302
   destination port: 80
2. acknowledgement number: 207
   source port: 80
   destination port: 302
3. acknowledgement number: 127

Problem 4

What are the three charms in *Harry Potter* that corresponds to our three project names? What are their effects, respectively?

1. Accio (Project 1)
2. Confundo (Project 2)
3. Riddikulus (Project 3)

1. "This charm summons an object to the caster, potentially over a significant distance."
2. "Causes the victim to become confused, befuddled, overly forgetful, and prone to follow simple orders without thinking about them."
3. "A spell used when fighting a Boggart, 'Riddikulus' forces the Boggart to take the appearance of an object upon which the caster is concentrating."

Source: `https://en.wikipedia.org/wiki/List_of_spells_in_Harry_Potter`