Consider the following example (see sem_ex.c):

```c
10 /* child process*/
11 if (fork() == 0) {
12     for (i = 0; i < nloop; i++) {
13         sem_wait(&mutex);
14         printf("child: %d
", (*ptr)++);
15         sem_post(&mutex);
16     }
17     exit(0);
18 }
19 else {
20     /* back to parent process */
21     for (i = 0; i < nloop; i++) {
22         sem_wait(&mutex);
23         printf("parent: %d
", (*ptr)++);
24         sem_post(&mutex);
25     }
26     exit(0);
27 }
```

**[Q 1]** What is the purpose of the semaphore? [Hint: think shared resources and synchronization]

**[Q 2]** For nloop = 3, the program generates

```
parent: 0
parent: 1
child: 2
child: 3
```

Explain why the processes seem to run sequentially without resource conflicts? Is the semaphore really required? [Hint: think process scheduling]

**[Q 3]** How do you modify the code to force the semaphore to play a role? [Hint: think relinquish CPU]

**[Q 4]** What is the effect of removing line 15 or line 24? Explain. [Hint: think traffic light jam]

**[Q 5]** What happens if nloop = 200? Explain. [Hint: see Q 2, gcc –lrt sem_ex.c]
[Problem # 2]

[Q 1] Explain the basic steps that two network nodes follow to communicate with each other [Hint: think DNS, ARP]

[Q 2] The following UDP message was captured on the network:

```
0x0000: c84c 7520 2948 0800 276d 051f 0800 4500
0x0010: 003a f118 4000 4011 87d3 c0a8 3737 4a7d
0x0020: 7f6a a910 3039 0026 c1fe 4973 6e27 7420
0x0030: 4350 5343 3231 3320 6c6f 7473 206f 6620
0x0040: 6675 6e3f 3f21 210a
```

Determine the following:

- Destination MAC address
- Source MAC address
- Source IP address
- Destination IP address
- Source Port Number
- Destination Port Number
- Destination Host Name
- UDP Message Length
- UDP Checksum
- Message

[Q 3] Who is the destination equipment manufacturer?

[Q 4] Where does the IP Header start?

[Hint: Ethernet Frame, IP Header, UDP Header, some fields are optional]

http://en.wikipedia.org/wiki/Ethernet_frame
http://en.wikipedia.org/wiki/IPv4
http://www.string-functions.com/hex-string.aspx
Search for c84c75 and 080027]