Computer Networking Assignment 6

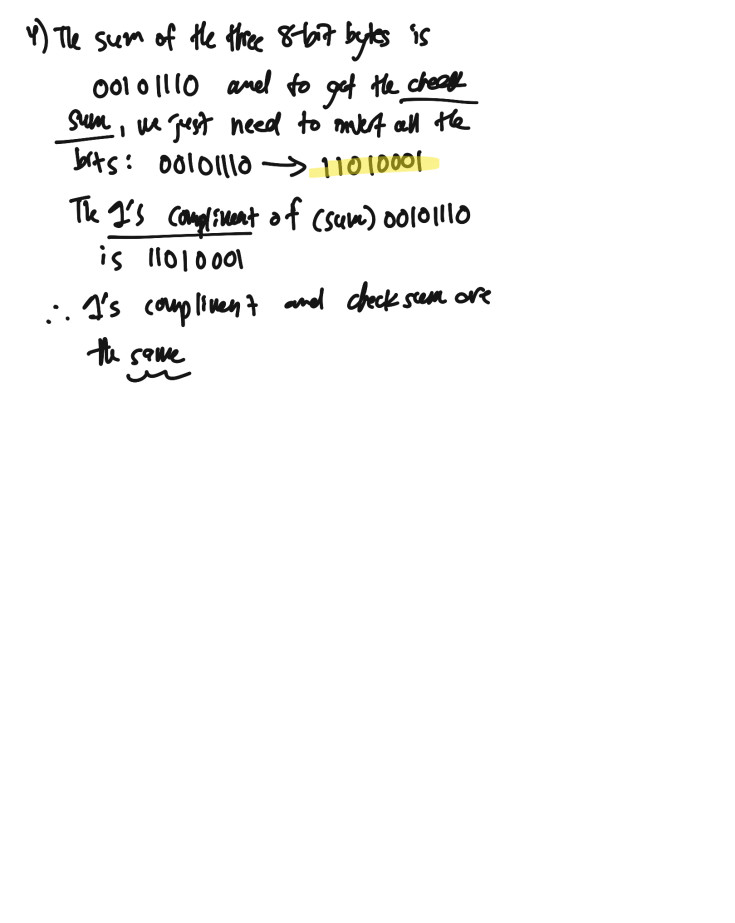
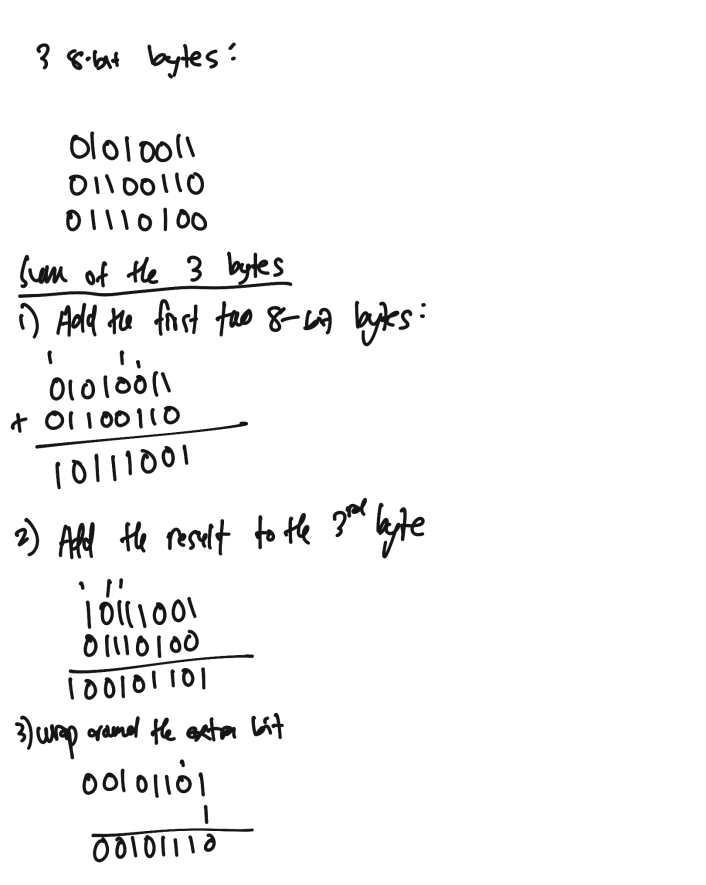
Due in 2 Weeks

**Part 1- Problems of Chapter 3 in Textbook**

P3:

UDP and TCP use 1s complement for their checksums. Suppose you have the following three 8-bit bytes: 01010011, 01100110, 01110100. What is the 1s complement of the sum of these 8-bit bytes? (Note that although UDP and TCP use 16-bit words in computing the checksum, for this problem you are being asked to consider 8-bit sums.) Show all work. Why is it that UDP takes the 1s complement of the sum; that is, why not just use the sum? With the 1s complement scheme, how does the receiver detect errors? Is it possible that a 1-bit error will go undetected? How about a 2-bit error?

**My Work**



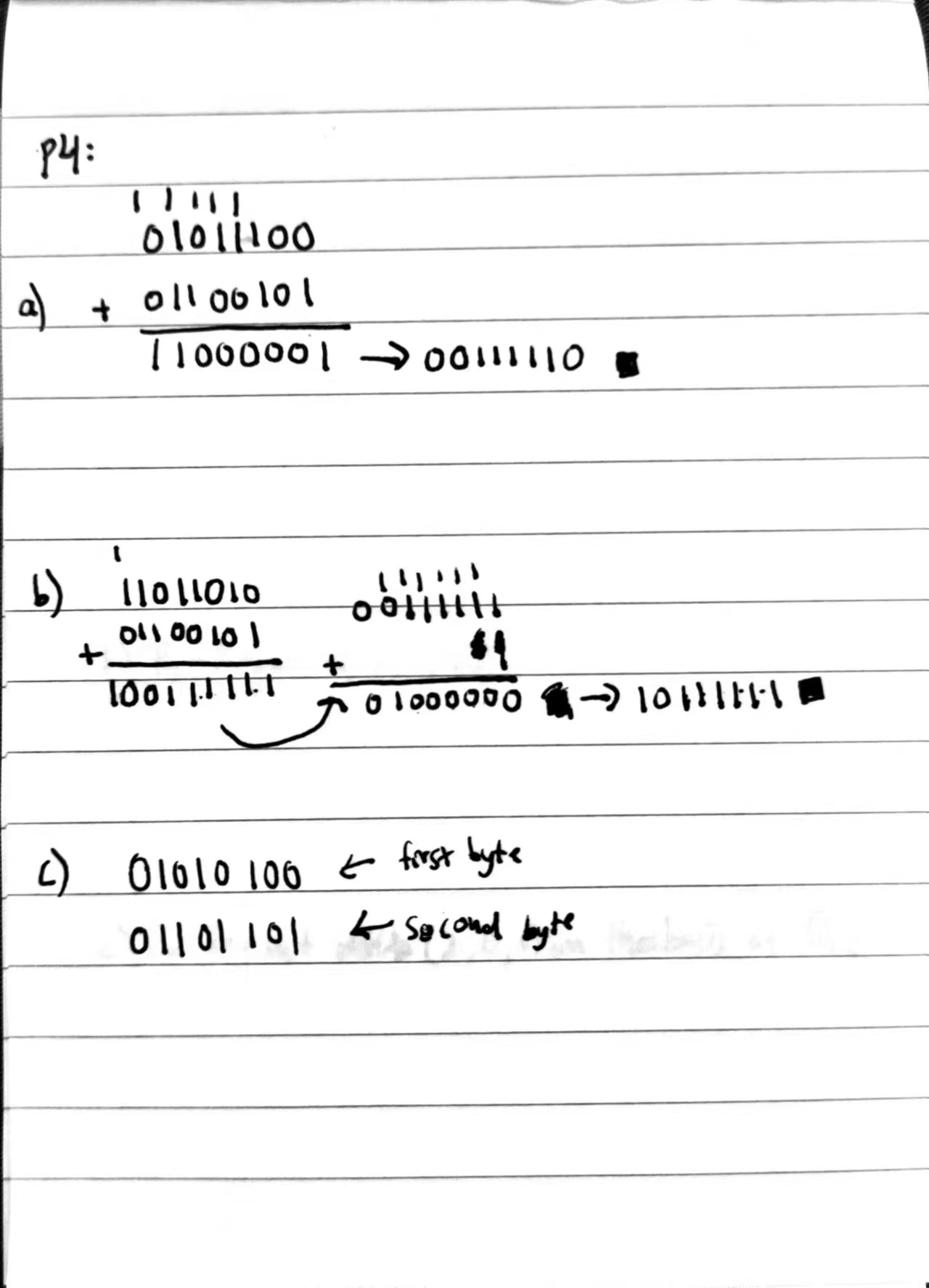
**User Datagram Protocol (UDP) cannot just use the sum. Rather, it has to use the checksum. UDP uses 1’s compliment because it is the same as the checksum of the sum. The checksum is the basically the invert of all the bits in the sum. The checksum is used by the receiver to identify the errors in a segment. To detect errors, the receiver has to first add all the bytes including checksum and then observe the sum. If it contains all 1’s, then it has no errors, but if it contains one or more 0’s, then the segment contains errors. By using 1’s complement, it is possible to detect all the 1-bit errors but there is a possibility that some 2-bit errors might be left undetected.**

P4:

1. Suppose you have the following 2 bytes: 01011100 and 01100101. What is the 1s complement of the sum of these 2 bytes?

b. Suppose you have the following 2 bytes: 11011010 and 01100101. What is the 1s complement of the sum of these 2 bytes?

1. For the bytes in part (a), give an example where one bit is flipped in each of the 2 bytes and yet the 1s complement doesn’t change.



**Part 2- Coding**

Use python to implement the Internet checksum, try to employ matplotlib for data visualization.

**import struct**

**import matplotlib.pyplot as plt**

**def internet\_checksum(data):**

**# Pad data if necessary (adds an extra zero byte to the data if its length is not even)**

**if len(data) % 2 != 0:**

**data += b'\x00'**

**# Calculate the checksum**

**checksum = 0**

**for i in range(0, len(data), 2):**

**word = data[i] + (data[i + 1] << 8)**

**checksum += word**

**checksum = (checksum & 0xffff) + (checksum >> 16)**

**# Return the checksum as a 16-bit unsigned integer**

**return struct.pack('!H', ~checksum & 0xffff)**

**# Example data**

**if \_\_name\_\_ == '\_\_main\_\_':**

**data = b'hello world'**

**# Calculate the checksum**

**checksum = internet\_checksum(data)**

**print(f"Checksum: {checksum.hex()}")**

**# Count the frequency of each byte**

**counts = [0] \* 256**

**for b in data:**

**counts[b] += 1**

**# Plot the histogram**

**plt.bar(range(256), counts)**

**plt.title("Byte Frequency Histogram")**

**plt.xlabel("Byte Value")**

**plt.ylabel("Frequency")**

**plt.show()**

