COMP 273: Assignment 1

Question 1

1. My binary data representation uses two bytes to represent a fraction. The first byte is used to represent the numerator and the second byte is used to represent the denominator. Additionally, the first bit of each byte is used to represent whether it is positive or not (0 for positive and 1 for negative). In this way, negative fractions can be represented and both parts of the fraction can be used for two’s complement arithmetic. To receive the decimal version of the fraction the computer must first divide the numerator and denominator. Even if both numerator and denominator are negative the value shouldn’t be affected because the returned value is a division of both integers.
2. To represent the binary value will be 00000101 00000111. Five is represented by the first 8 bits and 7 is represented by the last 8 bits. For a negative fraction such as the binary representation will be 10000011 00000101. The negative is represented by the first 1.
3. This binary data structure will take up 20 bytes worth of data if 10 fractions are made.
4. A large compromise made by this representation is the fact that the numerator and denominator can only representation integers from -127 to 127. This can be alleviated somewhat by allowing more bytes to represent the numerator and denominator but the representation will still be finite. In addition, in order to receive the decimal point version of the fraction the computer first must divide the numerator and denominator.

Question 2

1. **92A5F16**– binary: 1001 0010 1010 0101 11112  
    – decimal: **1000101101012** – decimal:   
    – hex: 8B516  
   **33910** – binary: 339/2 = 169R1, 169/2 = 84R1, 84/2 = 42R0, 42/2 = 21R0, 21/2 = 10R1, 10/2 = 5R0, 5/2 = 2R1, 2/2 = 1R0, 1/2 = 0R1  
    Answer: 1010100112  
    – hex: (Used binary solution) 15316
2. Assuming ASCII (all in binary):

|  |  |
| --- | --- |
| Address | Byte |
| 000100010000 | 00101001 |
| 000100010001 | 00100000 |
| 000100010010 | 01100001 |
| 000100010011 | 01101101 |
| 000100010100 | 00100000 |
| 000100010101 | 01001101 |
| 000100010110 | 01100001 |
| 000100010111 | 01110010 |
| 000100011000 | 01111001 |
| 000100011001 | 00101110 |

Question 3