CS19001 Programming and Data Structures Lab

Assignment Set 8 **May 23, 2023**

1. An ***Internet Protocol address*** (IP address) is a numerical label assigned to each device connected to a computer network that uses the Internet Protocol for communication. Internet Protocol version 4 (IPv4) defines an IP address as a 32-bit number. IP addresses are written and displayed in standard notations, such as 172.16.254.1, where the 32-bit number is broken into 4 bytes, separated by dots, and each byte is written in decimal. For example, the IP address 172.16.254.1 consists of the four decimal numbers, 172, 16, 254, and 1. The corresponding 32-bit binary address is:

   ```
   10101100 00010000 11111110 00000001
   ```

   For a hierarchical implementation of the internet, the IP address is viewed as a concatenation of a network address and a host address. For example, if the most significant 18 bits are treated as the network address, then the remaining 14 bits give us the host address. In this case, for the IP address 172.16.254.1, we have the following:

   ```
   IP Address:        10101100 00010000 11111110 00000001
   Network Address:   10101100 00010000 11000000 00000000
   Host Address:      00000000 00000000 00111110 00000001
   ```

   This network address (in standard notation) is 172.16.192.0, and the host address is 0.0.62.1.
   Write a program that does the following:
   (a) It reads an IP address (such as 144.16.192.44) and prints the binary 32-bit address with a blank space separating each byte. **[5 marks]**
   (b) It reads the network address (in standard notation) from the command-prompt, and then prints the network address and host address (in standard notation) corresponding to the IP address read earlier. **[5 marks]**

2. We wish to write a program which reads a set of complex numbers as command line arguments and prints their sum. The executable will be called **`cxadd`**. Some examples of the usage of this program is given below:

   ```
   Command line:      ./cxadd 2 + i5 + 3 + i7 + 9 + i3
   Expected output:   14 + i15
   ```

Command line: **./cxadd 2 – i8 + 3 + i7 - i3 + 2 - 5**

Expected output: **2 - 4i**

Note that in our case, a complex number may have both the imaginary and real parts, or only the real part, or only the imaginary part. Imaginary parts are prefixed with the letter *i*. You may assume that each plus or minus operator is preceded and succeeded by a blank character. Also, you may assume that each real or complex part has only one digit, i.e. "10" or "i15" are not allowed. Hint: you may find the C library function **`atoi()`** useful. **[5 marks]**

3. Write a C program to display (in binary) the sign and exponent fields of a double-precision floating point number read from the command-prompt. Name your executable **`showfields`**. Hint: you may find the C library function **`atof()`** useful. An example run is shown below:

   Command line: **./showfields 1.20**

   Output:

   Sign: 0

   Exponent: 01111111111

   [5 marks]