Network Applications Programming - Homework 2
(FTP program with data compression)

Motivation:

FTP is a common network service. Since you have learned how to write UNIX socket programs in class, this homework asks you to practice the socket programming skill.

Homework Content:

You need to provide the FTP service in the homework. By using TCP sockets, you can create a FTP server and a FTP client. Your FTP client has to support some commands:

- **connect** IP_addr port_num:
  Connect to a server. Your program should allow users to specify the IP address and port number of a server. Notice that the server side needs to print out the client's connection information (e.g., IP address, port #, socket type, etc.)
- **goodbye**:
  Terminate the connection. The server side has to also show the leave of the client.
- **upload** filename:
  Upload a file called "filename" to the server. Your FTP service should support non-textual files, for example, MP3 or WORD files. Beside, you need to provide file compression to save the transmission time. In particular, you can use fixed-length Huffman coding to compress the file. When you use the "upload" command, your client in fact should send two separated files to the server:
  1) The coded version of "filenanme".
  2) Necessary information related to Huffman coding (e.g., the appearing frequency of each alphabet in the file).

Here are some examples of the above commands:

Client side (140.113.1.1)

```
[student @ CSE ~ ]$ connect 140.113.1.2 3456
The server with IP "140.113.1.2" has accepted your connection.
[student @ CSE ~]$ upload 1.txt
Original file length: 140,234 bytes, compressed file length: 23,768 bytes (ratio: 16.95%)
Using fixed-length codeword (3 bits)
[student @ CSE ~]$ goodbye
See you next time.
```

Server side (140.113.1.2 port 1234)

```
[TA @ CSE ~] server # TA runs the server program
A client "140.113.1.1" has connected via port num 1732 using SOCK_STREAM (TCP)
The client sends a file "1.txt" with size of 140,234 bytes. The Huffman coding data are
```

```
stored in "1-code.txt".
# TA will open "1.txt" here to test its correctness. Also, TA will check the "1-code.txt" file
# to make sure that you indeed compress the file via Huffman coding.
The client "140.113.1.1" with port 4323 has terminated the connection.
```

Below gives some format of the Huffman coding-related information:

```
Fixed-length Huffman coding (3-bit codeword):
 Alphabet (ASCII code)       Frequency (%)             Codeword
 A (65)                      140,231 (21.45%)          000
 B (66)                      50,123 (7.67%)            001
 C (67)                      63,154 (9.66%)            010
 …
 Z (90)                      234,453 (35.86%)          011
```

Requirements:

- You need to use UNIX socket programming in this homework.
- You have to provide a makefile. TAs will deduct your grades if there is no makefile, or the makefile is erroneous.
- You must submit a README file along with your program. The README file should briefly describe how you write your codes (for example, the idea of your program).
- You have to demonstrate your program. TAs will announce the demonstration time.

Grading Policy:

You need to submit your codes and demonstrate your program to TAs. The due day of this homework is **4/15**. You will get **no point** if you do NOT demonstrate your program (even if you submit codes). Discussion among your classmates is encouraged. However, plagiarists will get **ZERO point**. Below are the points you can get in this homework:

Items	Points
Socket connection (i.e., connect and exit commands on the client side and the connection information on the server side)	10%
File transmission	20%
Huffman coding	50%
User interface (for example, welcome message or help command)	10%
Code's comments and README file	10%
Bonus: You ALSO provide variable-length Huffman coding. [Note] If you implement only variable-length Huffman coding (without the fixed-length one), you will NOT get this bonus.	20%

Reference: (Huffman coding) http://en.wikipedia.org/wiki/Huffman_coding