NETWORKS
PROJECT 3
Socket Programming: Cookies

1 A cookie server: Getting some wisdom

What we're after: As a first foray into programming with *sockets*, we will create a *fortune cookie* server (for brevity, dropping the *fortune* from the name). Specifically, our goal would be to create a *client/server pair* that behaves as follows:

- The client establishes a connection with the server.
- The server transmits a randomly chosen *fortune*—a message that, one hopes, conveys some wisdom—to the client.
- The client and server disconnect.
- The client displays the fortune.

How to get started: You will write your own Java code from scratch, creating two programs: `CookieServer` and `CookieClient`. The essential Java classes that you will need are the `Socket` and `ServerSocket` classes. They each contain a large number of methods, so here is a listing of the ones most relevant here:

- In `ServerSocket`, `ServerSocket (int port)`: The constructor for listening on the given `port` for a new connection. Remember that using this constructor only creates the socket; it does not itself listen for a connection.
- In `ServerSocket`, `accept()`: Listens for a connection, and once one is made, accepts it. This method returns a `Socket`.
- In `ServerSocket`, `close()`: Close all connections through the given socket and tear down any listeners.
- In `Socket`, `Socket(String host, int port)`: The constructor for initiating a connection to the given `host` on the given `port`.
- In `Socket`, `getInputStream()`: Returns an `InputStream` through which data can be read from the socket.
- In `Socket`, `getOutputStream()`: Returns an `OutputStream` through which data can be written to the socket.
- In `Socket`, `close()`: Close all connections through the given socket.

How your code should behave: In one window, you should be able to run the server, which will wait for a connection, provide a fortune, and then exit. Using it should look something like this:

```
(remus)$ java CookieServer 12345
Listening on port 12345...
Connection established
Fortune sent
Exiting
```

On the client side, which would need to be run in a separate terminal, using it should look like this:

```
(romulus)$ java CookieClient remus.amherst.edu 12345
Connecting to remus.amherst.edu:12345...
Connection established
Your fortune: To get the best grade, stop worrying
about grades. --Prof. Kaplan
Exiting
```

Your server should, ideally, have access to a collection of fortunes, and it should choose one of them at random to send. Where should you get the fortunes? I hear that the Internet may have some that you can use. Alternatively, you could make some up.

2 How to submit your work

Submit your `CookieClient.java` and `CookieServer.java` source code files (whichever one you chose to modify for this assignment), using one of the two usual tools:

- **Web-based:** Visit the submission system web page.
- **Command-line based:** Use the `cssubmit` command at the shell prompt on `remus/romulus/vega.cs`.

This assignment is due on Friday, Nov-16, 5:00 pm.