CSEE 4119 Spring 2016: Assignment 2 (Updated 02/15/2016)

This assignment contains a number of small programming problems, each requiring about 20-50 lines of code, depending on your style and language chosen. You can solve these problems in either C/C++, Java, Python, or PHP. Please avoid the use of web frameworks (Django, Flask, Ruby-on-Rails, etc.), as they obscure many of the lower-level behavior.

1. Modalities: Which interaction modalities are appropriate for the following applications:
   a) Find out today's weather;
   b) Be alerted when the temperate drops below freezing;
   c) Distribute video coverage of a Bernie Sanders (or Donald Trump, if you prefer) visit to Columbia University to 2,000 students;
   d) Update a web page when a computation completes.
2. *Port numbers*: Find out the port numbers for ftp, and the protocol used by the Nest thermostat.
3. *UTF-8*: Create a simple web application that detects whether the form input is a combination of letters and digits only (e.g., no symbols or spaces), using regular expressions and Unicode character classes. Your form can use either GET or POST (or both). Do not use Javascript on the client.
   a) The application should show a form and then display the UTF-8 string entered and whether it meets the letter-digit criterion.
   b) Also display the length of the string, both in bytes and in characters.
   c) Depending on the language preference of the browser, display an appropriate "welcome" message. (You only need to support English, Finnish and Korean, but can add others if you like.)
4. *UTF-8*: What is the sequence of Unicode characters (U+xxxx) and byte string (in hexadecimal) for the Unicode "word" consisting of:
   a) the Latin capital letter T;
   b) the Hebrew letter Bet;
   c) the Euro sign;
   d) the CJK ideograph 木;
   e) the heart emoji.
5. *Serialization*: Create an XML and JSON representation of a simple family tree, with each member of the family having a name, birth year and year of death. We'll use the composer family Bach (https://en.wikipedia.org/wiki/Bach_family) as our example. Your representation should also include marriage relationships. Validate your XML and JSON using the W3C and a JSON validator (e.g., http://jsonlint.com/), respectively. Parent-child relationships should be represented by hierarchy where possible, not identifiers. You can find an introduction to XML at http://www.sitepoint.com/really-good-introduction-xml/ .
6. *Cookies:* Set and display a cookie entered by the user via a form input. (You can probably re-use some of the code from problem 1.) You should have two web pages: one that allows the user to enter the text string, and the other to show the cookie value returned

by the browser. Python offers the Cookie module (`import Cookie`); PHP uses the `setcookie()` function. In general, you can use the

```
Set-Cookie: name=value
```

`HTTP header format.`

7. *HTTP:* For each of the following HTTP scenarios, calculate the number of round trip times (RTT) required to fetch and receive an index.html file with seven embedded jpeg images (J1, ..., J7). Assume all the images and the index.html file are individually (but not together!) small enough to fit in one packet. Include the RTT for any TCP connection setup, but don't worry about the TCP connection closing time.
   a) HTTP 1.0 with no parallel TCP connections;
   b) HTTP 1.0 using up to three parallel TCP connections (your solution should minimize the number of total RTTs);
   c) HTTP 1.1 with persistent connections and no pipelining
   d) HTTP 1.1 with persistent connections and pipelining.

8. *HTTP/2.*0: Using the Akamai HTTP/2.0 demo page at https://http2.akamai.com/demo and the Chrome or Firefox developer tools, explain the difference in delay for the spinning globes. (Hint: To simplify the inspection, you can open each frame in a separate tab by using "Open Link in New Tab" when mousing over HTTP/1.1 or HTTP/2.) (a) How many simultaneous connections are used in each case? (b) Can you explain why the HTTP/2.0 download completes sooner?