CSCI-4220 Network Programming - Spring 2016
Homework 5: TLS HTTPS Client

**Your solution must be in C or C++ using the sockets API and OpenSSL as discussed in class.
Please name your file <your_rcs>_hw5.zip (e.g. armena2_hw5.zip). .tar.gz is also OK.
Please include a plain text README file in your submission - 10pts penalty if missing
(markdown or other formats that can be read with “cat README” are also acceptable)**

For this assignment, you will be extending your program from Homework 3 to include support for https URLs. In the case where an HTTPS URL is given, your program should use OpenSSL to establish a secure connection to the server. In the case where a standard HTTP URL is given, an HTTPS connection should **not** be attempted. **If no port number is explicitly given in the URL, use port 80 for HTTP and port 443 for HTTPS. Code reuse is important here** - that is, you should use the same routines to parse the headers for both HTTP and HTTPS connections, instead of copying and pasting the same code in two places. One way to accomplish this could be to use a polymorphic “connection” class with read and write member functions. If you prefer to use C instead of C++, note that something like C++’s virtual functions can be implemented in C using structs and function pointers:

```
struct connection {
        union {
                BIO *bio;
                int sockfd;
        }
        ssize_t (*read)(struct connection *conn, void *buf, size_t bytes);
        ssize_t (*write)(struct connection *conn, const void *buf, size_t bytes);
};
```

Create a connection struct, then fill in either bio or sockfd. Write 4 wrapper functions: one for read(), one for write(), one for BIO_read(), and one for BIO_write(). For an SSL connection, fill in conn.read and conn.write with the BIO wrappers, and for a normal connection, use the read()/write() wrappers. You may want to add an additional function to shut down the connection. You can call the “member” functions as conn->read(conn, buf, size) etc. **This is a suggested implementation and not a requirement - you are welcome to use C++ classes and virtual functions, or any other language features, to implement both HTTP and HTTPS.**

You may use the Connection: close request header so that the server will close the connection after the complete content has been sent.