Jordan Mai
1645861

1. **Project Description:**
    a. Httpserver is a program that is supposed to receive and send bytes that was delivered from a client server. It connects to a port with the client and once the client sends something, the server's job is to perform the task if possible and send back a response code. The program should send tasks such as sending over files contents and possibly writing from a file. A response code will also be sent to the client to send a message. The program will end once tasks are completed or if manually closed by the user.

2. **Program Logic - PSEUDOCODE**
    a. Handle_connection
        i. While (able to receive bytes from client curl):
            1. Parse each word separated by space. To obtain the request, body of file, version, content length and content. Then store each in a separate buffer.
            2. IF(request is "GET"):
                a. Open body file for reading
                b. Read content into the created file
                c. Create a response header and send required code to the client.
            3. IF(request is "PUT"):
                a. Open body file for reading if possible, else create new file
                b. Read in the content to a buffer
                c. Created a response header with content of file at the end
                d. Send to client to be written into their buffer
            4. IF(request is "HEAD"):
                a. Repeat "GET" but only send header and not body for response
        ii. Close(connection)

3. **Data Structures**
    a. **Array-** An array is used to store some byte amount to be used. We will use an array buffer to store bytes taken in from the client and also use arrays to write information into.
    b. **Files -** A file is used to put bytes of content into to be kept outside of a program or in this case to be sent over to users. For this program, a file is used to read and write bytes from the client server.

4. **Functions**
    a. **Int main:**
        i. This function opens a socket and attempts to connect to the same port as a client server. This function checks for any errors when reading from argv[] and makes sure that all arguments are correct. It will persist of a while loop to keep waiting and receiving connection until closing.
    b. **Strtouint16(char numer[]):**
        i. Converts a string to a 16 bit unsigned integer
        ii. Return 0 if string is out of range
    c. **create _listen_socket(unit16_t port):**
        i. Creates a socket for listening to the port
        ii. Closes and creates an error message on failure
    d. **Handle_connection(int connfd):**
        i. This function receives every byte that is sent from the client server. It will continue to keep receiving. This function will handle all necessary requests (“GET”, “PUT”, “HEAD”) only and send back a response to the client. THis function will parse through the entire buffer and take out necessary elements in different buffers. For a “GET” response, the function will read in bytes from a file if possible and construct a header response by selecting the correct response code and then to be sent to the client. For a “PUT” response, the function will create the intended file to be read and written to the client’s requested file, and then construct a header response. For a “HEAD” response, it will act the same as a “GET” but only returns the header and not the body. The function will keep receiving information until close.
5. **Question**