Nicholas LaJoie, ECE 331, HW 11

```
// Author: Nicholas LaJoie
// ECE 331 - Homework 11
// Date: April 20, 2017

1. Sockets in C

    NOTE: I was unable to complete this problem (ran out of time), so I still need to look int
o fetching an
    entire webpage. However, I did get a good start at understanding the syntax/form of socket
 programming
    in C. It's easier to see, however, why we would just do it in python instead.

    a) Tutorial: www.thegeekstuff.com/2011/12/c-socket-programming/

    b) Source Code

// Author: Nicholas LaJoie
// ECE 331 - Homework 11, Problem 1
// April 18, 2017
// C Program that fetches a webpage using a TCP Client
// Source: www.thegeekstuff.com/2011/12/c-socket-programming/
// NOTE: This code does not do what was requested on the homework (fetch a full web page)
// I still need to take time to figure that out...

#include <sys/socket.h>
#include <sys/types.h>
#include <netinet/in.h>
#include <netdb.h>
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <unistd.h>
#include <errno.h>
#include <arpa/inet.h>

int main(int argc, char * argv[])
{
    // Variable declarations - use sockaddr_in struct
    int sockfd = 0, n = 0;
    char recvBuff[1024];
    struct sockaddr_in serv_addr;

    // Takes ip address at the command line
    if (argc != 2) {
        printf("\n Usage: %s <ip of server> \n",argv[0]);
        return 1;
    }

    // Allocate receive buffer
    memset(recvBuff, '0',sizeof(recvBuff));

    // Attempt to create the socket connection
    if((sockfd = socket(AF_INET, SOCK_STREAM, 0)) < 0) {
        printf("\n Error : Could not create socket \n");
        return 2;
    }

    // Allocate server address structure memory
    memset(&serv_addr, '0', sizeof(serv_addr));

    // sockaddr_in struct - assign address family (what type of addresses you can communicate
with)
    // and assign port by using htons() to convert from host byte order to network byte order
```

Nicholas LaJoie, ECE 331, HW 11

```
    serv_addr.sin_family = AF_INET;
    serv_addr.sin_port = htons(5000);

    // Convert source address string to network address structure of type AF_INET
    if(inet_pton(AF_INET, argv[1], &serv_addr.sin_addr) <= 0) {
        printf("\n inet_pton error occured\n");
        return 3;
    }

    // Make the connection!
    if( connect(sockfd, (struct sockaddr *)&serv_addr, sizeof(serv_addr)) < 0) {
        printf("\n Error : Connect Failed \n");
        return 4;
    }

    // Read info from the socket
    while ((n = read(sockfd, recvBuff, sizeof(recvBuff)-1)) > 0) {
        recvBuff[n] = 0;
        if(fputs(recvBuff, stdout) == EOF) {
            printf("\n Error : Fputs error\n");
        }
    }

    // If the index becomes negative, something went wrong while reading
    if(n < 0) {
        printf("\n Read error \n");
    }

    return 0;
}
```

2. Python fetch - I chose to fetch the google homepage

```
#!/usr/bin/python3

import urllib.request

print(urllib.request.urlopen("http://www.google.com").read())
```

```
************************
* wc:  5     4     105 *
************************
```

3. See attached