client.c readme

client.c consists of two functions and a main

- int getSignal(char* data)
- int transmitSignal()

<u>Summary</u>

The job of the getSignal function is to gather the decibel value of the laptop and fill in the allocated data variable on the heap. The transmitSignal function sets up the network connection with sockets to the sever, calls the getSignal function to get the decibel value, and sends the decibel value over the signal. The whole transmitSignal function is in a while loop in main so it continuously runs and allows the server to continuously pinpoint user location.

int getSignal(char* data)

char* data represents a pointer to a dynamically allocated piece of memory on the heap done in the transmitSignal() function

system("iwconfig wlo1>netinfo.txt")

This outputs all information about the wireless interface wlo1 in a text file names netinfo.txt. Different computers will have different interface cards and will be called different.

On my HP laptop it is wlo1, this code can be made more modular my having a configuration file to enter all information about the computer. (This feature is not included) You can figure out your computer's interface name by ifconfig

system("grep 'level=' netinfo.txt | sed 's/^.*: //'>signal.txt");

The grep and sed command finds the line with the decibel range from the previous created text netinfo.txt and extracts the line and outputs to signal.txt

Note: I found this grep command on the web on sites like stackoverflow, unix, or etc. I cannot find the exact site where this command was found in, but it seems common as others sites have many different variations of this command. Also of course I added the two text files and keyword to look for and placed it in the system(…).

```
FILE *fp;
fp = fopen("signal.txt", "r");
char signalLine[110];           //stores line in array
fgets(signalLine,100,fp);
char c = '-';    //token to find
char* val = strchr(signalLine,c);    //finding location of -
```

This chunk of code opens up a file descriptor for the signal.txt text file and extracts it into an array signalLine. Then we look for the token '-' using the strchr method on signalLine(this indicates the start of the decibel value). Finally we get a char pointer to point to that location.

```
memcpy(data,val,4);
```

This copy copies 4 bytes of memory from val to data.

Val is the char pointer we designated with the strchr to point to the location of the character that is the start of the decibel value.

Data points to a dynamically allocated memory on the heap we did in the transmitSignal method.

```
fclose(fp);
```

We have to close the open file pointer when we are done using it.

```
int transmitSignal()
```

Note: This method contains socket programming. Large chunks of this code is taken directly from video tutorials I have used to learn socket programming. Each link is a video in the playlist. The first link will bring you to the playlist and video 1.

https://www.youtube.com/watch?v=eVYsIolL2gE&list=PL0JmC-T2nhdgJ2Lw5YdufR8MffaQdAvEf
https://www.youtube.com/watch?v=xfRdYrQUQeQ&list=PL0JmC-T2nhdgJ2Lw5YdufR8MffaQdAvEf&index=2
https://www.youtube.com/watch?v=d9pmc7oObkw&index=3&list=PL0JmC-T2nhdgJ2Lw5YdufR8MffaQdAvEf
https://www.youtube.com/watch?v=0XHIUgMVyV4&list=PL0JmC-T2nhdgJ2Lw5YdufR8MffaQdAvEf&index=4
https://www.youtube.com/watch?v=iSWrInT8CDs&list=PL0JmC-T2nhdgJ2Lw5YdufR8MffaQdAvEf&index=5
https://www.youtube.com/watch?v=pTYNQwWqB2Y&list=PL0JmC-T2nhdgJ2Lw5YdufR8MffaQdAvEf&index=6
https://www.youtube.com/watch?v=FRm9nk9ooC8&index=7&list=PL0JmC-T2nhdgJ2Lw5YdufR8MffaQdAvEf

```
int streamLength = 7;          //size to send over network
```

This indicated the length of the stream to send over the network

```
char* data = (char*)calloc(streamLength, sizeof(char));          //will hold wifi decibels
```

This is where we dynamically allocated the space in the heap to use later in the getSignal function to hold the decibel value

```
struct sockaddr_in remote_server;          //will hold ip info of server
int sock;                                  //socket to access server
 if((sock = socket(AF_INET, SOCK_STREAM, 0))==-1){
                perror("socket: ");
                exit(-1);
```

```
        }
remote_server.sin_family = AF_INET;
remote_server.sin_port = htons(25000);    //remote server port number
remote_server.sin_addr.s_addr = inet_addr("127.0.0.1");         //remote server ip
bzero(&remote_server.sin_zero, 8);
```

This chunk of code sets up a socket to send data over the network. We are using the sockaddr_in struct to input ip settings of the remote server. When we use the connect method we will have to zero out the server.sin_zero portion of the struct and cast it to a sockaddr pointer type. We do this because all the portions of the sockaddr is not needed for our local ip communication. Also we use the sockaddr_in structure first because it is easier to input data into this struct and then cast it.

Because of different endianness of different computer we have to use htons to change from host byte ordering to network byte ordering

We made a design decision to use TCP so that is why we are using SOCK_STREAM

```
connect(sock,(struct sockaddr*)&remote_server, sizeof(struct sockaddr_in));
```

This establishes connection to the server.

```
 getSignal(data);
```

This is where we fill the data variable with our decibel value to send over the network to the server.

```
if((send(sock, data, streamLength, 0) )==-1){                   //transmitting data over network
                perror("send error");
                exit(-1);
        }
        free(data);
        close(sock);
        sleep(2);
```

Here we do the actual sending across the network with the previously inputted streamLength.

We have to free the allocated memory as this code is run in an infinite while loop and would cause the computer to run out of memory real fast without it. We also do not want to keep file descriptors open.

Sleep is used because there is no need to update the location of the user less than every 2 seconds. This limits the probability of network saturation. This number is a design decision we took.