1) Open Cygwin or command prompt (make sure you have environmental variables set to compile C programs)
2) Change directory to directory of server.c and client.c
3) Compile and run the server.c using gcc server.c and ./a.exe (Windows)
4) Compile and run the client.c using gcc client.c and ./a.exe (Windows)
5) To change the hash function simple change 0 to 1 in server.c. In the following example I am using my own hash function.

```
#include <netdb.h>
#define HASHASCII 0 //for ascii hash 0 for ignore 1 for if u wanna implement
#define WORDSUM 0
#define MYHASH 1
static char word[200][200]; //for storing words
static char hash[200][200]; //for storing hashes
int numberOfWords = 0; //serves as index
```

6) Type the tweet (in client console) you want to encode: Example: I am a good boy.
7) The result would be in the format: 0x ___ ____ ____ ___ ____
8) To decode: Type in this format: 0x ___ ___. Example: 0x 324 432 144

MyHash function: It uses the ascii values and multiply it by 57. Then increment 57 by 1. And sum everything up. This way great will not be same as grate.

```
Multiplying 103 of g with number 57 to get 5871
Multiplying 114 of r with number 58 to get 6612
Multiplying 101 of e with number 59 to get 5959
Multiplying 97 of a with number 60 to get 5820
Multiplying 116 of t with number 61 to get 7076
```

I am using TCP because I don't want to deal with loss packets and my program is running pretty fast in TCP so I don't really need more speed. TCP is more reliable. My program is session oriented to make it easier for user as he can decode whole sentence at once. Transaction oriented would have been easier for me but more difficult for user. The decoding is differentiated from encoding by 0x in the start of message. No protocol information is kept on either side. I will implement file I/O to keep track on hashes and words. In ASCII hash function if both words have same encoding, it would save both words in word array and would store same encoding in hash array. To decode, the word which is stored first in array would be returned. WordSum and MyHash can't have same encoding. If the server doesn't respond or data packet is lost, it would just display relevant error message : such as "Error in connect(), Error in receive()" so you would need to restart both client and server.
Testing was done at my home using my own laptop so client and server on the same machine. I will try different machines in the demo. In case if the hash doesn't exists or you type hash to decode right after starting the server, the server will reply with 0x only. Past tweets are lost if the server is restarted. I will probably use file I/O to save the past tweets in file and read from file once the server is started. Initial testing was done uptil 10 words in each array.
It works but there can be some improvements made:
Improvements: past tweets, more reliable hash function, redundant code avoidance, make functions of hashes instead of everything one main.

Bonus: I can't encode the tweet right now but I know how it works. I have included a small program (decode.c) and a picture explaining how it works. I have encoded: "Mary had a little lamb". Just change the array on top word by word to get the encoding. I just need to use bruteforce to decode tweet which I will do later on.