Assignment 2, CS641, Due Date: Wednesday, Oct 5 2018, 5pm
Zip all your code as a single file and upload to Moodle.

In this assignment you must build a P2P network using socket programs with the following characteristics.

1. *Seed information:* On launching the client at any node, it finds out the IP addresses of one or more other nodes which we call *seed nodes*. You could hard code a URL of a web page which has this information, or use any other means to get this seed information.

2. *Learning peer IP addresses from seed nodes:* From the learned IP addresses of the seed nodes, the client chooses at most 3 seed nodes to setup TCP connections with. It then downloads the list of peers which each of these seed nodes is aware of. You will have to use some fixed port numbers for your application so that the client knows which port to connect to. You should handle the case of some seed nodes being offline, that is they do not respond to TCP connection requests. The client then closes the TCP connections to the seed nodes. The seed node adds the IP address of the client to its list of "peers".

3. *Connect to peers:* From the list of IP addresses learned (the seed nodes and their peers), at most 4 are chosen randomly to become *peers* of the client. How you generate pseudo-randomness, for the random selection, is your choice. A TCP connection is setup with each peer.

4. *Broadcast:* Any peer can generate a string of the form "`<timestamp>: <MyIP> pays <N> BTC to <RandomPeerIP>`". You have to replace all elements in `<>` with suitable IP addresses and numbers. Such messages can be generated once every 5 seconds. A node generating a message transmits it to all its peers. Every node maintains a data structure with the SHA-1 hash of every message already forwarded by it to all peers. When any message is received, it takes its SHA-1 hash and checks this data structure for a match to see if it has forwarded the message already. If it has not, then it forwards it to all the peers it is connected to. If it has already finished sending it to all peers, it discards the message.

5. *Write to file*: The first time a node receives a message, it writes the string to an output file along with the IP address of the node from which it received the message. The format of each line of the file will be:
`<timestamp>: <MyIP> pays <N> BTC to <RandomPeerIP>: RECEIVED FROM <PeerIPAddress>`