Group Member: (Student Name, Student Number)
Rui Yang, 467656
Can Zhu, 467711

First of all, we will discuss how two nodes connect to each other. In our design of this P2P application, each node is considered both as the client and server. If one node (named A) wants to connect another node (Named B), two connections should be established. Firstly, A has to send a join request to the B. When B received the request, B can determine whether to establish the connection or not. In our scenario, all incoming connections are accepted unless the connected number exceeds the pre-defined number of the maximum allowed connections (Note: those two connections between A and B are considered as one in this case). At this time, A acts as the client and B as the server. A can send request to B. But B cannot initiatively send request to A. It only responses to the request from A. So, in order to establish a "complete" connection between A and B, B has to send a join request to A also. Once these two connections have been establish. Both A and B then are considered as fully functioned node. Basically one node should maintain two tables to record the related information about the connected nodes, one for incoming connection and another for outgoing connection. When a connection has been established, IP of the remote node, port of the remote node, socket that used to communicate the remote node and a valid flag used to record the status of the connection should be stored into the corresponding table. If one connection broken, the valid flag should be set to false.

Then, we will present how to implement these sub-functions. When the Node application started, a thread named "server_handler" will be created to handle incoming TCP connection. And another thread called "heart_beat" will be created to send connected nodes with PingA message per 5s. Then the workflow goes into the console, which is infinite loop to handle commands from the end user. The console mainly accept 5 commands:

- help -- To display all acceptable commands;
- quit -- To quit the application;
- join ip -- To join one node with the IP of the remote node;
- pingA ip -- Carrying out A category ping to other node;
- pingB ip -- Carrying out B category ping to other node;
- Bye ip -- To disconnect from other node;
- query key -- To query key;
- listInNode -- To list all incoming connections;
- listOutNode -- To list all outgoing connections.

In sum, when a node is up, there are mainly three workflows: "heart_beat", the console and "server_handler". Note: when a new TCP request is accepted, the "server_handler" will create a "connection_handler" thread to handler requests

from this newly connected node. So basically, “connection_handler” acts as the server and also the intermediary among the P2P network.
(The formats of all messages strictly follow the formats that are given in the assignment sheet.)