Names: Christian Mitton (cem249), Abhishek Patel (asp238)

1.) Briefly discuss how you implemented your recursive client functionality.

As instructed, python2.7 was used. To discuss how the recursive client functionality was implemented, the project can be split into 3 parts: rs.py, ts.py and client.py.

client.py

client.py accepts 3 arguments: the hostname of the machine where rs.py is running, the port number of rs.py and the port number of ts.py. It is assumed that the following would be typed on the terminal: *$ python2.7 client.py <rsHostname> <rs_port_number> <ts_port_number>*. It is assumed that a file called PROJI-HNS.txt will contain the hostnames for testing, the DNS table for rs.py will be in PROJI-DNSRS.txt, the DNS table for ts.py will be in PROJI-DNSTS.txt, and that a file called RESOLVED.txt will be used to check the resulting output. With each execution of client.py, the contents of RESOLVED.txt will be overwritten.

Once ran, client.py will send each hostname in the PROJI-HNS.txt to rs.py. The rsHostname and port number is used to connect to the machine running rs.py. For each hostname, a response from rs.py is received. If the hostname has been found in the RS DNS table, the response will be the hostname's corresponding entry, and this is written to RESOLVED.txt from client.py.

If the hostname hasn't been found, <tsHostname> - NS is the response sent to client. Client.py would then connect to ts.py using this tsHostname and the *ts_port_number* that was passed as an argument. It is assumed that the tsHostname will be included in PROJI-DNSRS.txt with the flag 'NS'. If the hostname is in the TS DNS table, its corresponding entry will be sent back to client, and then be written to RESOLVED.txt. If the hostname is not in the TS DNS table, the hostname along with " - ERROR:HOST NOT FOUND" is sent back to client, and this is written to RESOLVED.txt.

rs.py

Once this server is set up on a given machine using the specified port number, it forever listens for incoming messages, and each message is assumed to be a hostname from

client.py. Once the message is received, the DNS table from PROJI-DNSRS.txt is checked to see if the hostname exists in it. If it does, the entry in PROJI-DNSRS.txt corresponding to the hostname is sent back to client. If not, the tsHostname is retrieved from PROJI-DNSRS.txt, and <tsHostname> - NS is sent to client.

ts.py

Once this server is set up on a given machine using the specified port number, it forever listens for incoming messages, and each message is assumed to be a hostname from client. Once the message is received, the DNS table from PROJI-DNSTS.txt is checked to see if the hostname exists in it. If it does, the entry in PROJI-DNSTS.txt corresponding to the hostname is sent back to client. If not, the hostname along with "- ERROR:HOST NOT FOUND" is sent back to client.

2.) Are there known issues or functions that aren't working currently in your attached code? If so, explain.

There are no known issues or functions that aren't working.

3.) What problems did you face developing code for this project?

Understanding how to get the hostname of a machine, how to connect to a provided hostname, and interweaving this functionality with the specifications of the project description was a bit confusing at first. But after many tests, everything is working.

4.) What did you learn by working on this project?

It gave a greater understanding of how multiple servers can work with each other to produce a desired output. It also gave a glimpse into the complexities behind the communication and functionality of Recursive DNS clients and DNS servers, which is a valuable insight to have as Computer Science majors.