Napster-style Peer-To-Peer File Sharing System

Output File

CS-550 Advanced Operating Systems

Name: Suraj Kumar Didwania (A20334147)

Name: Lawrence Amadi (A20382063)

Project Title: Napster-Style Peer-to-Peer File Sharing System

OUTPUT FILE:

Step-by-Step Execution:

Server end:

A) The server end is running from workspace bin.

Started rmiregistry 5000 for the server using the command

$ start rmiregistry 5000

```
C:\Users\Suraj Didwania\workspace\NapsterP2P\bin>start rmiregistry 5000

C:\Users\Suraj Didwania\workspace\NapsterP2P\bin>
```

Creating server stub using rmic.

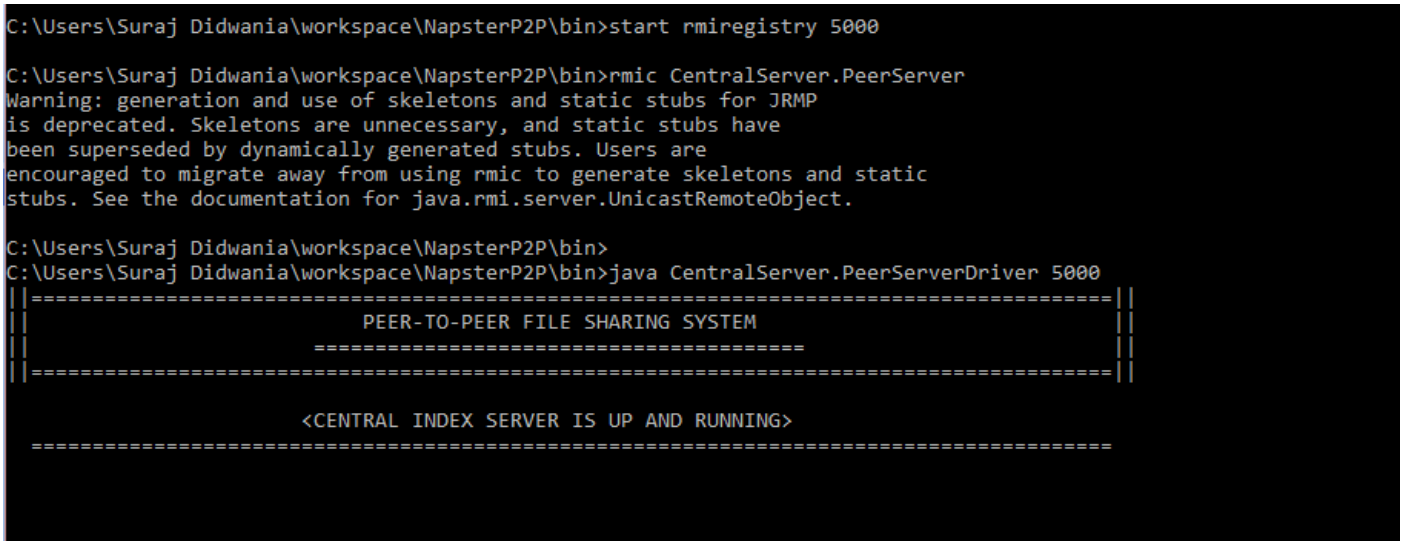
$ rmic CentralServer.PeerServer

```
C:\Users\Suraj Didwania\workspace\NapsterP2P\bin>rmic CentralServer.PeerServer
Warning: generation and use of skeletons and static stubs for JRMP
is deprecated. Skeletons are unnecessary, and static stubs have
been superseded by dynamically generated stubs. Users are
encouraged to migrate away from using rmic to generate skeletons and static
stubs. See the documentation for java.rmi.server.UnicastRemoteObject.

C:\Users\Suraj Didwania\workspace\NapsterP2P\bin>
C:\Users\Suraj Didwania\workspace\NapsterP2P\bin>
```

Running central server using below command

$ java CentralServer.PeerServerDriver 5000

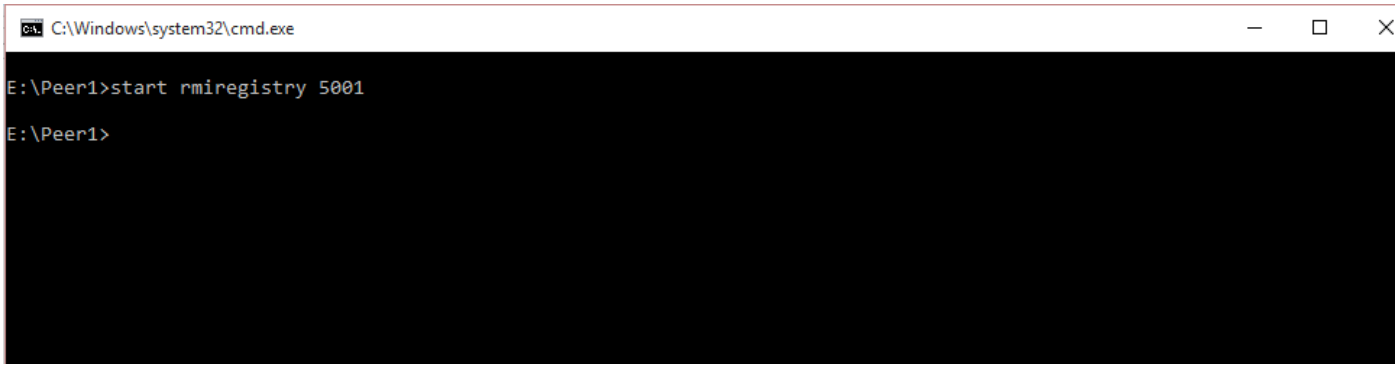


B) Peer1 end:

It is running from remote location and all the test files has been kept in the same folder.

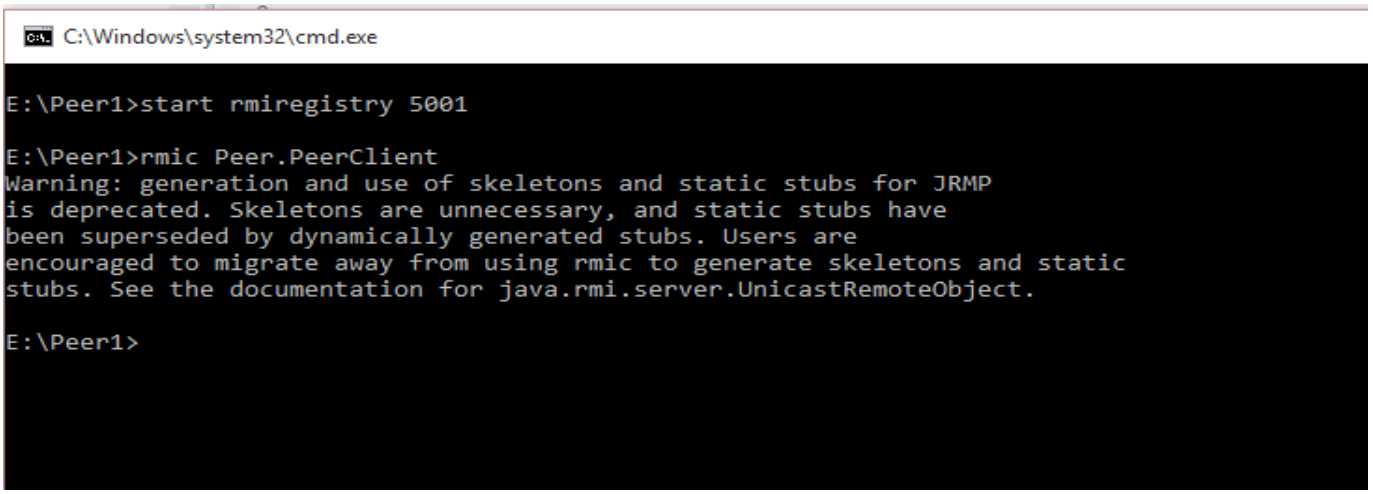
Started rmiregistry 5001 for the Peer1 using the command

$ start rmiregistry 5001



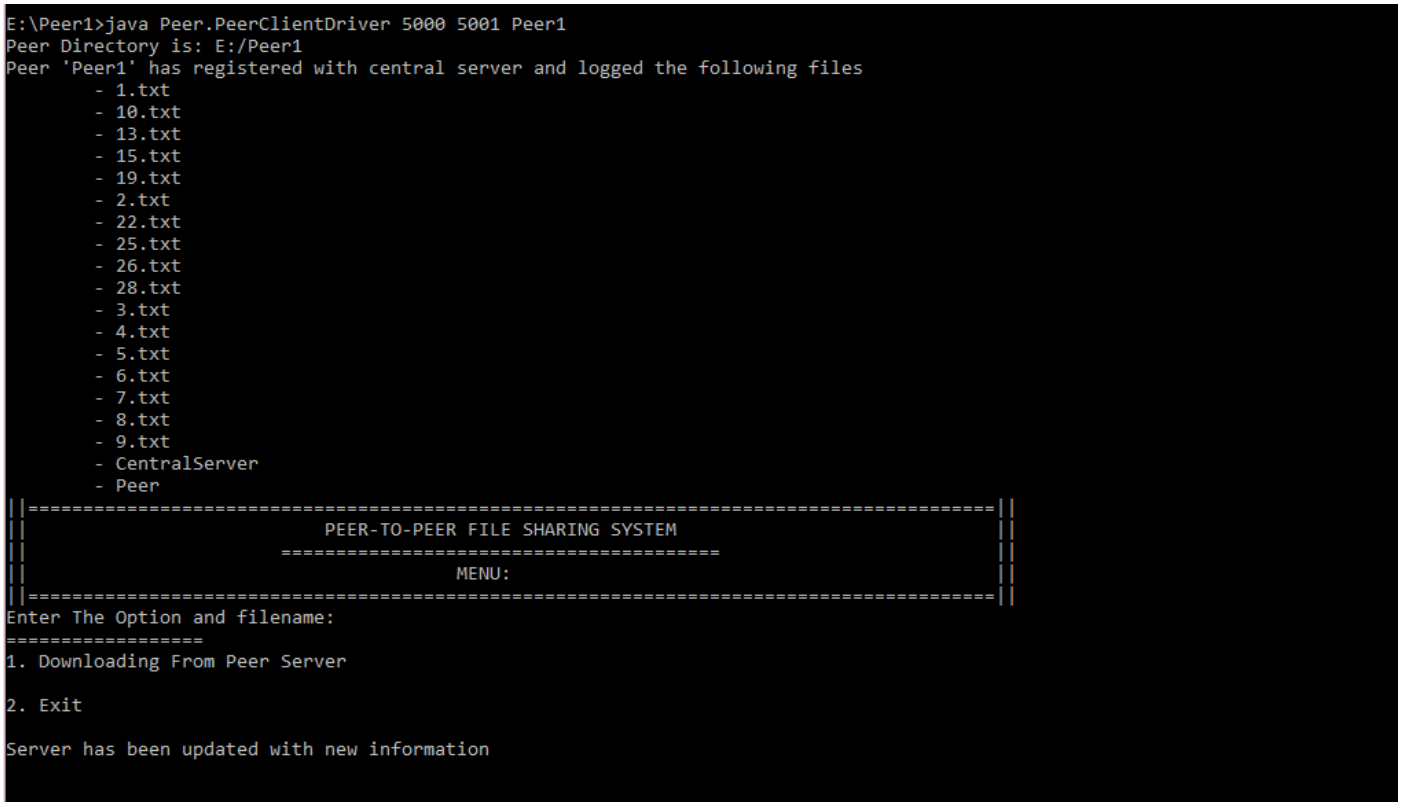
Running stub at the peer location using below command

$ rmic Peer.PeerClient



Running peer1 and registering all the files.

$ java Peer.PeerClientDriver 5000 5001 Peer1



C) Peer2 End:

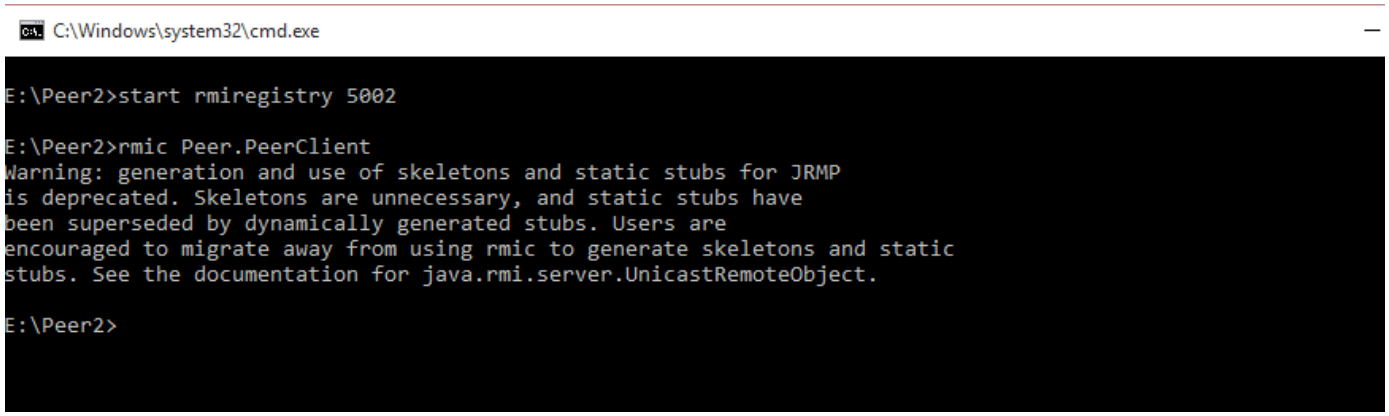
It is running from remote location and all the test files have been kept in the same folder.

Started rmiregistry 5002 for the Peer2 using the command

$ start rmiregistry 5002

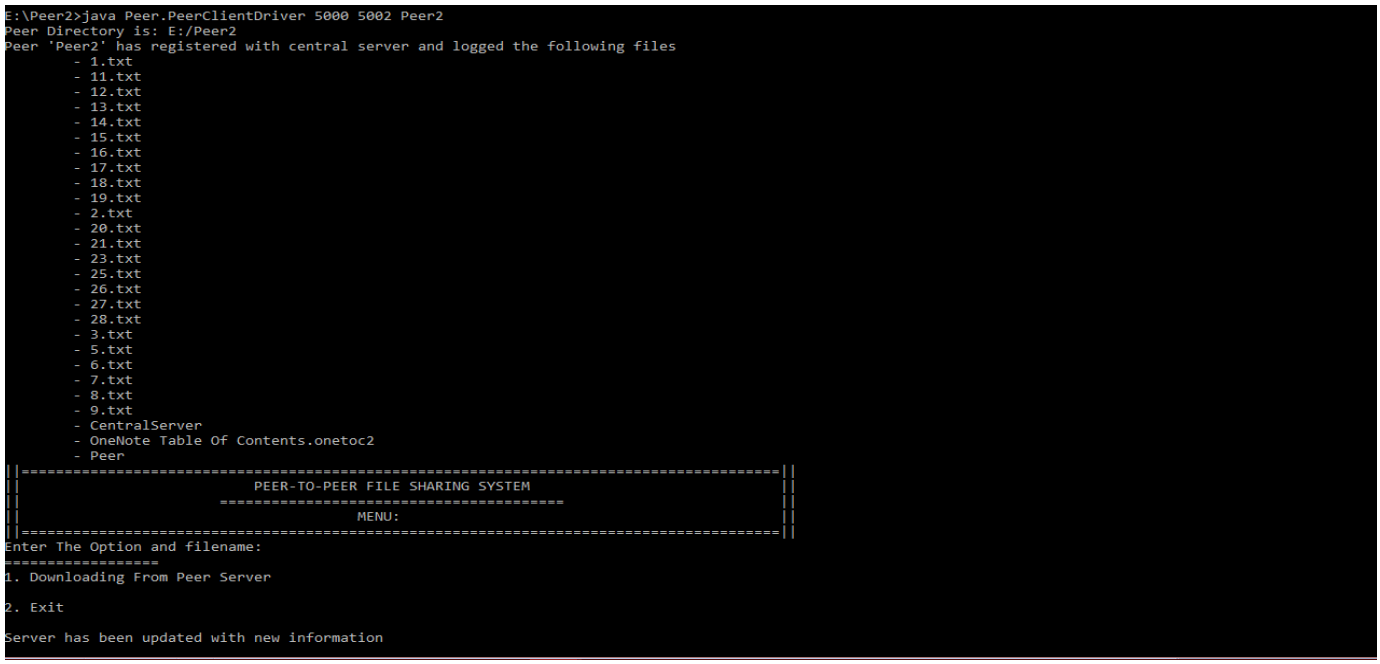
Running stub at the peer location using below command

$ rmic Peer.PeerClient



Running peer2 and registering all the files.

$ java Peer.PeerClientDriver 5000 5002 Peer2



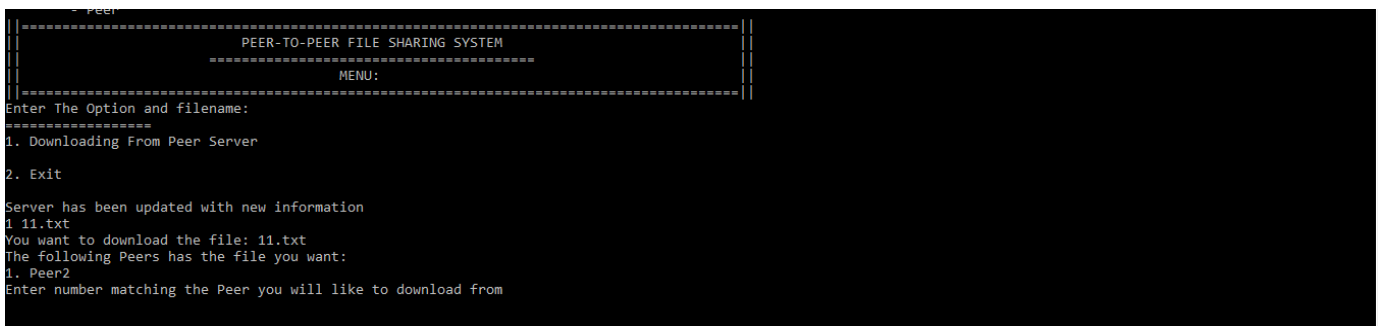
So in the server end all the files have been registered and index has been stored.

Downloading the files:

Peer1 wants to download the file 11.txt method so the peer has to give

$ 1 11.txt

The no of peers which contains the files will be shown below.



```
Enter number matching the Peer you will like to download from
1
File downloading!!
File has been downloaded
Server has been updated with new information
Average response time of the Peer Peer1 is 1.687ms
```

File has been downloaded after giving the peer option and kept in the peers location.

If the server gives the file name which he already has, it will validate and ask to prompt again.

```
Server has been updated with new information
File '11.txt' has been sent to Requesting Peer: Peer1
1 2.txt
Please enter the filename which you don't possess
```

After the peer2 has given the detail about the peer name, the file gets downloaded and kept in peer2 location.

```
1 10.txt
You want to download the file: 10.txt
The following Peers has the file you want:
1. Peer1
Enter number matching the Peer you will like to download from
1
File downloading!!
File has been downloaded
Server has been updated with new information
Average response time of the Peer Peer2 is 1.099ms
```