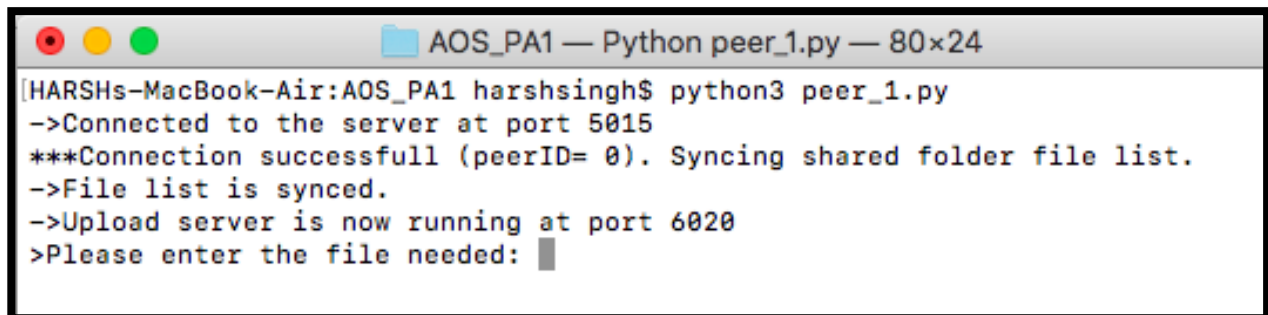


**Napster Style Peer to Peer File Sharing System**  
**CS550- Advanced Operating System**  
**Harsh Singh (A20398109)**

## Output File.

### Case 1: File download Scenario:

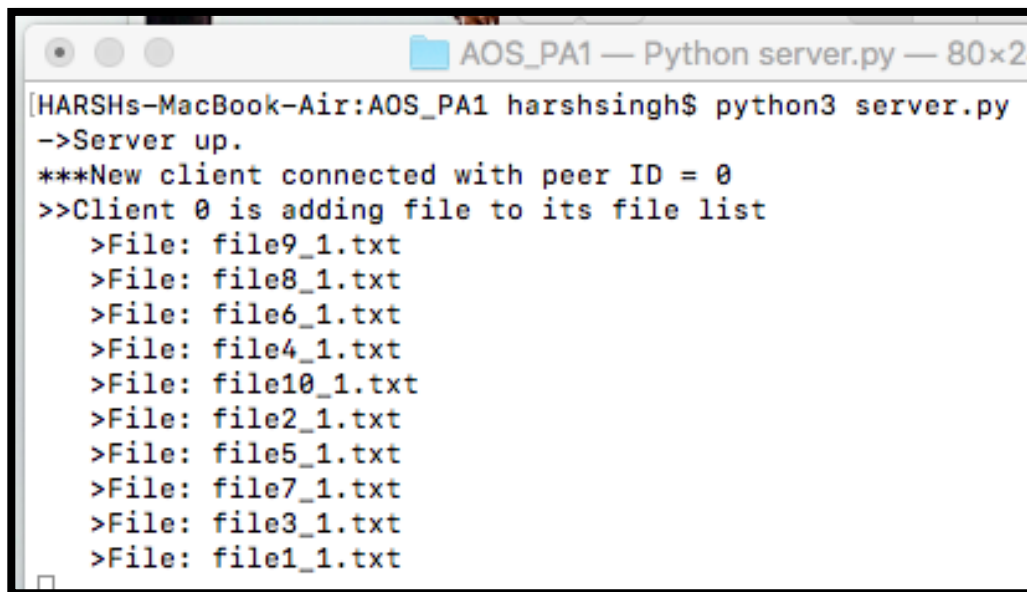
- a) After the initial setup (peer ID allotment, file list sync and running server for inter peer communication... see figure 1), peer application prompts user if he/she wants to search and download any file.



```
[HARSHs-MacBook-Air:AOS_PA1 harshsingh$ python3 peer_1.py
->Connected to the server at port 5015
***Connection successfull (peerID= 0). Syncing shared folder file list.
->File list is synced.
->Upload server is now running at port 6020
>Please enter the file needed: █
```

Figure 1 Peer 1's console after initial routine

\*After the initial setup stage and first peer connection server looks like in figure 2. It gets all the file list in the peer shared directory

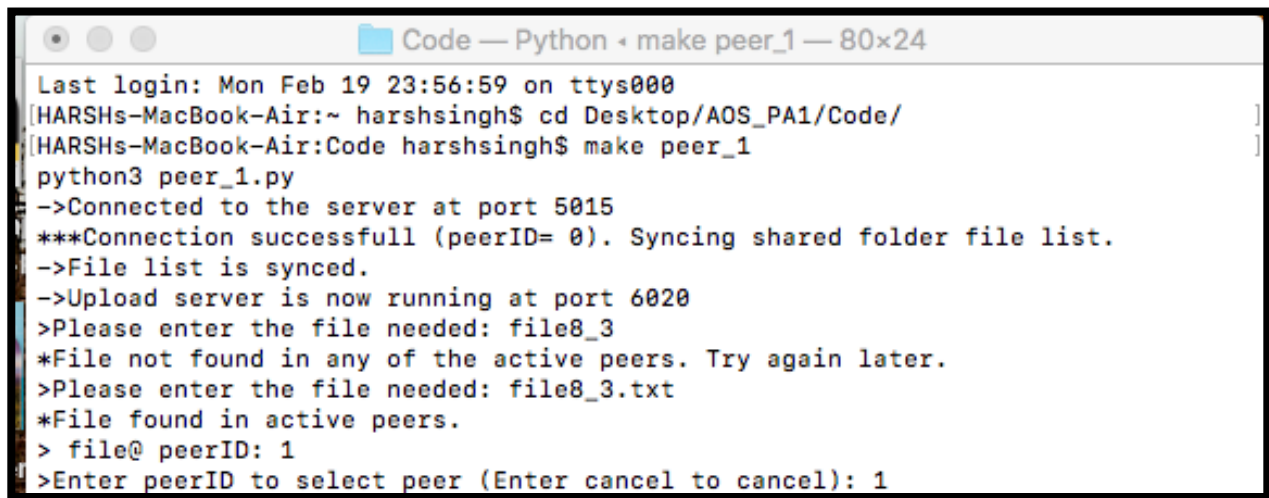


```
[HARSHs-MacBook-Air:AOS_PA1 harshsingh$ python3 server.py
->Server up.
***New client connected with peer ID = 0
>>Client 0 is adding file to its file list
>File: file9_1.txt
>File: file8_1.txt
>File: file6_1.txt
>File: file4_1.txt
>File: file10_1.txt
>File: file2_1.txt
>File: file5_1.txt
>File: file7_1.txt
>File: file3_1.txt
>File: file1_1.txt
```

Figure 2 Indexing Server's console after the initial routine

- b) As soon as user enters his/her desired file, server sends port and peer ID of the peer holding the file. It will return all the peers having that file. It can be seen in figure 3 that if file is not available with any of the peer then, it prompts user to try again.

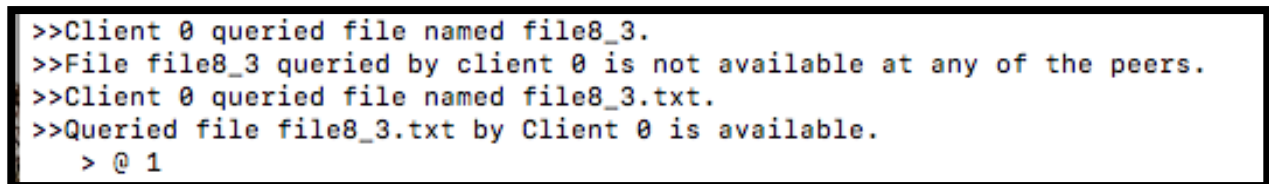
If file is available then, user is asked to choose from all possible peers holding the file. User gets 3 chances. To Select the peer, peerID is the key shown by the application. In this case, 1.



```
Code — Python • make peer_1 — 80x24
Last login: Mon Feb 19 23:56:59 on ttys000
[HARSHs-MacBook-Air:~ harshsingh$ cd Desktop/AOS_PA1/Code/
[HARSHs-MacBook-Air:Code harshsingh$ make peer_1
python3 peer_1.py
->Connected to the server at port 5015
***Connection successfull (peerID= 0). Syncing shared folder file list.
->File list is synced.
->Upload server is now running at port 6020
>Please enter the file needed: file8_3
*File not found in any of the active peers. Try again later.
>Please enter the file needed: file8_3.txt
*File found in active peers.
> file@ peerID: 1
>Enter peerID to select peer (Enter cancel to cancel): 1
```

Figure 3 User prompted by peer application to select

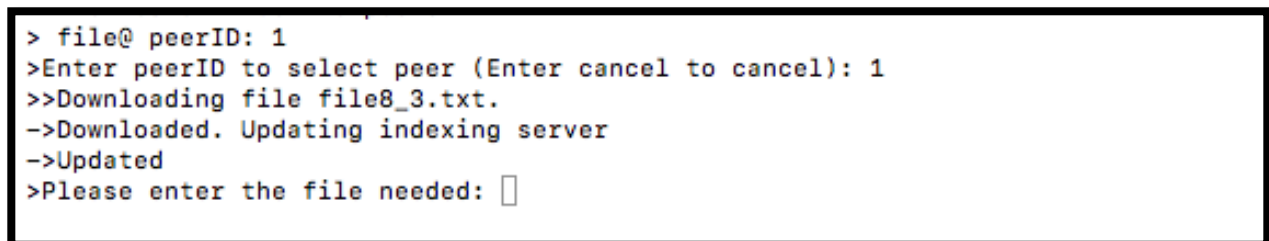
\*server side looks like figure 4. It can be noted that peer received the first request and shows file was not available. When user queried another file that exists in some peer, it returned the peerID



```
>>Client 0 queried file named file8_3.
>>File file8_3 queried by client 0 is not available at any of the peers.
>>Client 0 queried file named file8_3.txt.
>>Queried file file8_3.txt by Client 0 is available.
> @ 1
```

Figure 4 Server Side: event with respect to peer's request

- c) After user selects the peer, download starts, if no error occurs. Peer connects to other peer's server, downloads the file, closes the connection. Also, the new file downloaded gets registered with the indexing server.



```
> file@ peerID: 1
>Enter peerID to select peer (Enter cancel to cancel): 1
>>Downloading file file8_3.txt.
->Downloaded. Updating indexing server
->Updated
>Please enter the file needed: 
```

Figure 5 Peer downloads the file and updates the server

\*at indexing server side, peer updates the client list.

```

>>Queried file file8_3.txt by Client 0 is available.
> @ 1
>>Client 0 is adding file to its file list
>File: file9_1.txt
>File: file8_1.txt
>File: file8_3.txt
>File: file6_1.txt
>File: file4_1.txt
>File: file10_1.txt
>File: file2_1.txt
>File: file5_1.txt
>File: file7_1.txt
>File: file3_1.txt
>File: file1_1.txt

```

Figure 6 Indexing server receives new list of files

\*at another peer server, the other user gets message and the file gets uploaded. Finally, connection is closed.

```

HARSHs-MacBook-Air:Code harshsingh$ python3 peer_3.py
->Connected to the server at port 5015
***Connection successfull (peerID= 1). Syncing shared folder file list.
->File list is synced.
->Upload server is now running at port 6022
>Please enter the file needed:
>>File 0 requested by client file8_3.txt.
>>File file8_3.txt requested by client 0 is available and preparing to send.
>>File file8_3.txt requested by client 0 was sent.
->Disconnecting client with ID 0
>>Please enter the file needed:

```

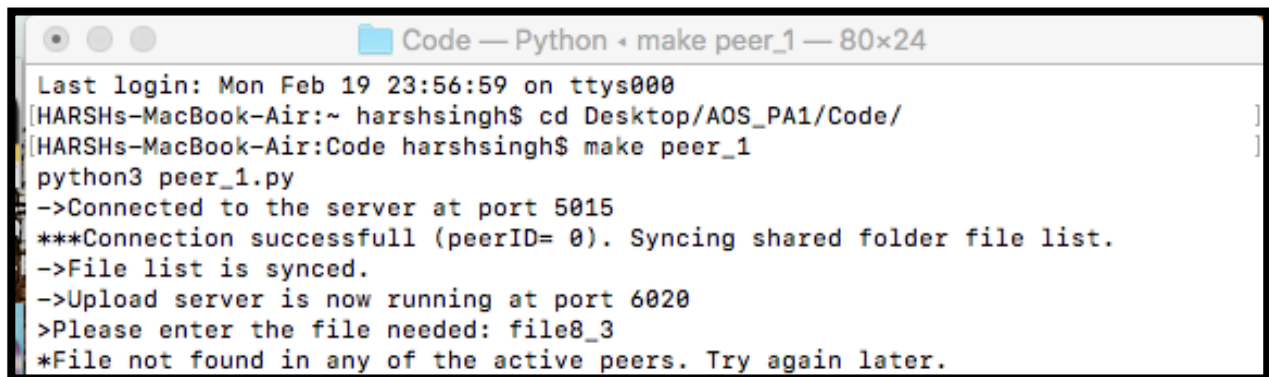
Figure 7 Console output of Peer holding the file.

## Case 2: Query Scenario:

Similar to previous case, peers will connect to server and send their file list in their shared directory to server. Server will allot ID to these connected peers and construct data structure to hold file list against the ID.

- a) In figure 8, user was prompted to input the file name, he/she wants to search. First, user asked for a file that was not available in any of the active peers. Hence, system responded by saying file is not available.

After that, user was again prompted to input the file name he/she wants. This time, user asked for a file available with peer with ID 1 (figure 9).



```
Code — Python • make peer_1 — 80x24
Last login: Mon Feb 19 23:56:59 on ttys000
[HARSHs-MacBook-Air:~ harshsingh$ cd Desktop/AOS_PA1/Code/
[HARSHs-MacBook-Air:Code harshsingh$ make peer_1
python3 peer_1.py
->Connected to the server at port 5015
***Connection successfull (peerID= 0). Syncing shared folder file list.
->File list is synced.
->Upload server is now running at port 6020
>Please enter the file needed: file8_3
*File not found in any of the active peers. Try again later.
```

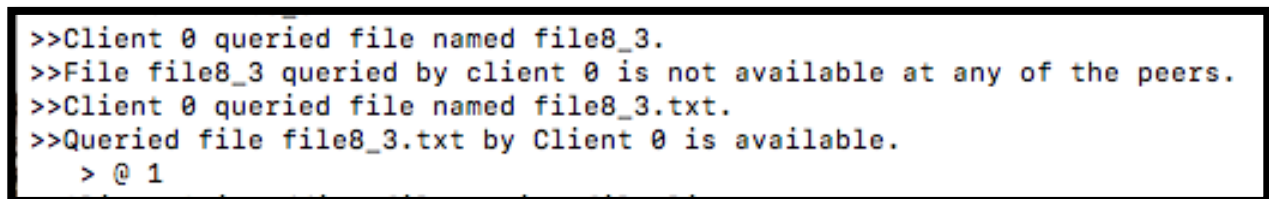
Figure 8 user putting file name.



```
Code — Python • make peer_1 — 80x24
Last login: Mon Feb 19 23:56:59 on ttys000
[HARSHs-MacBook-Air:~ harshsingh$ cd Desktop/AOS_PA1/Code/
[HARSHs-MacBook-Air:Code harshsingh$ make peer_1
python3 peer_1.py
->Connected to the server at port 5015
***Connection successfull (peerID= 0). Syncing shared folder file list.
->File list is synced.
->Upload server is now running at port 6020
>Please enter the file needed: file8_3
*File not found in any of the active peers. Try again later.
>Please enter the file needed: file8_3.txt
*File found in active peers.
> file@ peerID: 1
```

Figure 9 User searches for file and gets response.

At the indexing server, all the queries are recorded and logged into the console.



```
>>Client 0 queried file named file8_3.
>>File file8_3 queried by client 0 is not available at any of the peers.
>>Client 0 queried file named file8_3.txt.
>>Queried file file8_3.txt by Client 0 is available.
> @ 1
```

Figure 10 Indexing server console shows event of file search

```
>Please enter the file needed: file8_3.txt
*File found in active peers.
> file@ peerID: 1
>Enter peerID to select peer (Enter cancel to cancel):
```

Figure 11 User prompted to choose a peer holding the file.

ID of the peer becomes the selector. User gets 3 chances to select the other peer. After user selects, file gets transferred and updates server

```
>>Downloading file file8_3.txt.
->Downloaded. Updating indexing server
->Updated
>Please enter the file needed: 
```

Figure 12 After selection peer application shows further events.

```
>>Queried file file8_3.txt by Client 0 is available.
> @ 1
>>Client 0 is adding file to its file list
>File: file9_1.txt
>File: file8_1.txt
>File: file8_3.txt
>File: file6_1.txt
>File: file4_1.txt
>File: file10_1.txt
>File: file2_1.txt
>File: file5_1.txt
>File: file7_1.txt
>File: file3_1.txt
>File: file1_1.txt
```

Figure 13 Indexing server prints events onto the console.

The peer holding the file, receives the request and prompts its user about the event. It sends the file and disconnects itself.

```
HARSHs-MacBook-Air:Code harshsingh$ python3 peer_3.py
->Connected to the server at port 5015
***Connection successfull (peerID= 1). Syncing shared folder file list.
->File list is synced.
->Upload server is now running at port 6022
>Please enter the file needed:
>>File 0 requested by client file8_3.txt.
>>File file8_3.txt requested by client 0 is available and preparing to send.
>>File file8_3.txt requested by client 0 was sent.
->Disconnecting client with ID 0
>>Please enter the file needed:
□
```

Figure 14 Other Peer server showing other user about the events.

**Problem that could occur:**

1. File name length: If the length exceeds certain length, server will only receive partial name. (a very unlikely scenario)
2. Port number are fixed hence, if port is busy, inter peer servers will not work. User will need to change port in params.py.