Name: Gaurav Desai
G Number: G00851337
TCP:

- TCPServer.java:

    - This java file has two classes one TCPServer and other ClientSocketThreadUtility. TCPServer starts the execution with the main method. To handle concurrency Selector and channels from java.nio API is used. For children port and client port one ServerSocketChannel is being created and registered with the selector. Selector has configured to have operation OP_ACCEPT which will the basis for selection among several channel means whenever a new channel is ready to be acceptable it will be selected by the Selector.select() call.

    - Selected Channel is received by iterating over SelectionKeys and we will get Socket from the ServerSocketChannel. Now for every socket and hence for every childrenport and clientport and also for multiple requests on those ports a new thread will be generated. This Thread routine is written in ClientSocketThreadUtility Class.

    - In the thread routine(run method), logic is written from receiving the search query sent form the client to finally giving the search results back to the same client with the direct results from this server's phonebook or getting results from parent server traversing through the vertically distributed server tree. It will first search the record in phonebook file for this server and it wont find a match it will forward the same request to parent and so on same way described in the handout.

    - It will print all necessary informative messages.

- TCPClient.java:

    - Client will connect with above server with the help of port specified in command line argument. I will ask the user to enter search keyword in following format:

```
--------------------------------------------------------------------------------

@TCP Client: Enter the query in following format or type 'exit' to exit:

             Format: '<First Name><space><Last Name>' (you can use '*' as wildcard
                     character for first name or last name):

             Search: scott heidi

             Search results for the keyword 'scott heidi':

             ---------------------
             First Name: SCOTT
             Last Name : HEIDI
             Year      : 4378
             ---------------------
--------------------------------------------------------------------------------
```

UDP:

- UDPServer.java:
    - The program logic and workflow is almost same as TCPServer mentioned above. Mainly, Selectors and channel logics differs. Here DatagramChannel is used. Now this channel has only to supported operations. So as can be seen in the code while registering selector for this channel,

      ```
      datagramChannel.register(selector, SelectionKey.OP_READ |
      SelectionKey.OP_WRITE);
      ```

    - This means that selector will make the channel available as it is ready for reading or writing. This is made it difficult for me to make the it work fully. Actually here it is repeatedly finding the channel readable and entering the loop and creating thread which I didn't want but somehow I could not find any solution. I am curious to know the solution for this and correct implementation of UDP.

    - Here also for thread routine is written in separate class ClientDatagramThreadUtility. This is also almost same as TCP part except the read write part for which ByteBuffer of java.nio API is used.
- UDPClient.java:
    - Client will connect with above server with the help of port specified in command line argument. I will ask the user to enter search keyword in following format:

```
--------------------------------------------------------------------------------

@UDP Client: Enter the query in following format or type 'exit' to exit:

             Format: '<First Name><space><Last Name>' (you can use '*' as wildcard
                     character for first name or last name):

             Search: scott heidi

             Search results for the keyword 'scott heidi':

             ---------------------
             First Name: SCOTT
             Last Name : HEIDI
             Year      : 4378
             ---------------------
--------------------------------------------------------------------------------
```