Data Services Code Challenge Design

- This Server Application is composed of three parts all within DataServiceServer.java:
    - **ServerStart()** is used to start the server. It initializes variables and objects that will be used by itself, closeAll(), ServerThread's, and TimingThread. It opens the TCP connection on port 4000, creates a FileWriter that writes locally to numbers.log in src/main/resources. The server then waits for a client to connect where it creates a new ServerThread(client socket)
    - **ServerThread** (sub class) is a thread that is created for every client that connects to the TCP socket with a max of 5. The FileWriter is accessible and used by all ServerThreads. There is a semaphore in place that only allows one thread at a time to write a number to the file as well as edit byte[] array used for de-duplication assurance. . To ensure that no duplicates are written I create a Byte array with size [125000000] where each bit represents the corresponding number of 0-999999999. I use this method for two reasons: Memory wise it only allocates roughly 125mb, and the lookup to check if a number has been written yet is Big O(1). The method uses (number/8) to select which byte and (number%8) to select which bit. From there if the bit is 0 the number is written and the bit is flipped. Alternatively, if the bit is already 1 it is not written to the file and is counted as a duplicate.
    - **TimingThread**(sub class) this is thread that is created immediately upon starting the server. This thread is responsible for outputting the ten second report. Before it reports is calls semaphore.acquire() so that no information is miscounted. After reporting the semaphore is released where it falls into a ten second timeout where it will repeat this process when it leaves that timeout.
    - **closeAll()** is called by the first ServerThread that receives a 'terminate' from a client. It waits for all threads to finish their current inputFromClient then closes the fileWriter and the serverSocket. After that it prints out the final counts and exits the program.

Assumptions

- The server will only close when "terminate" is sent through the connection
- All other assumptions made clear in problem description