Assignment 1
Internet Programming
SUID - 211778351

1. Servers are supposed to run for a long time without stopping, therefore they must be designed to provide good service no matter what their clients do. Determine anything that a client might do to cause it to give poor service to other clients. Suggest Improvements to fix the problems that you find.

Repeated Data Sent – A client can be identified as a bad client if it tries to send the same data again and again.
Solution – The modified bullet proof server will take care of this fact that no client should be allowed to send the same data again and again. In my bullet proof server, if the client tries to send the same data more than 5 times, it is identified as a bad client and its connection is terminated.

Multiple Clients - In the current echo client server code, it is not possible to run multiple clients at the same time. The server is designed in such a way that it is giving the service to only one client at a time. But we might want multiple clients to connect to the server at the same time, so this care has to be taken, that the connection to one client should not stop other clients from getting the service from the server.
Solution – We want our architecture to support multiple clients at the same time. For this reason, we should use threads on the server side, so that whenever a client request comes, a separate thread can be assigned for handling each request.

Packet Size - There should be some check on the size of the packet sent by the client (Like in the Skype Application). A bad client can send really large sized packets to the server which must not be allowed. Such packet must be discarded.
Solution – No client should be allowed to send the data more than the decided packet length. I have set the limit of 64 bytes on the server side to demonstrate this. Which means, if the client tries to send the packet whose size is more than 64 bytes, that client is identified as a bad client and its connection Is terminated.

Number of Client Connections – Let's suppose, multiple clients try to connect to the server, and none of these clients' close their socket connection. We can soon run out of the ports and then server won't be able to give service to more client.
Solution – To take care of this issue, at the server side, if any client stays connected longer, (without closing its socket connection), then the time out exception (InterruptedIOException) will be thrown and that client will be disconnected.

2. Based on your above answers revise TCPEchoClient.java to create such a client, which I call a **bad client**. Your bad client should attempt to cause the server to provide poor services to other clients, in at least 2 different ways. (2 points, 1 for each bad behavior).

With this pdf, I have submitted following 3 .java files for Client.
TCPEchoClientLongPacket.java[Command Line Arguments – 127.0.0.1 HiKomalfdjhkjhkjdhjkshjkdfhkjhgjkhsjkfhsdmnmvxncmvbuhjfdzbvmnbdvmnfbmnmdnfbdmbmndbfnclckxkldbmnnbn 7]
TCPEchoClientSocketNotClosed.java [Command Line Arguments – 127.0.0.1 Komal 7]
TCPEchoClientGUI.java [Command Line Arguments - 127.0.0.1 7]

To see that there are problems in the above files, try to run the above clients with the **TCPEchoServer.java** [Command Line Arguments - 7] in the following manner –

- You will notice that, TCPEchoClientLongPacket.java, which sends data (more than 64 bytes) is allowed by the current non-bullet proof server.

- Also, if you try to start more than one client program at one time, the current non-bullet proof server only keeps giving the service to one dedicated client, and it Is not able to accept the connections from rest of the clients which are trying to connect to this server.

- Any client in this case, can flood the receiver buffer by sending the same data (repeated data) again and again.

3. Based on your answers above, revise TCPEchoServer.java to incorporate those improvements. You may call this improved server a **bullet-proof server**. (2 points, 1 for each improvement)

BulletProofTCPEchoServer.java [Command Line Arguments - 7]
Suggestions –
Please find the **BulletProofTCPEchoServer.java** file attached with this pdf.
- I have enforced a check that number of bytes in a packet should not exceed 64 bytes. If a packet greater than 64 bytes arrive at server side, then this client is identified as a bad client and it is disconnected. You can also check this by giving very long string in the GUI Client. It will emit an error message in this case.

- Also, each client in the bullet proof server runs on different dedicated thread. Thus now, multiple clients can talk to one server at the same time and bullet proof server can serve the request of all the incoming clients.

- Also, two variables prevString and newString are used, which keep the track of the incoming messages to the server. If the same message is received more than 5 times, then that client is identified as a bad client and such client is disconnected.