Threads and Socket Programming

1. Create a Java application capable of creating multiple threads. (Use Runnable interface/Thread class).
2. Implement producer consumer problem in Java

**Problem Description**

There are two processes, a producer and a consumer, that share a common buffer with a limited size. The producer “produces” data and stores it in the buffer, and the consumer “consumes” the data, removing it from the buffer. Having two processes that run in parallel, we need to make sure that the producer will not put new data in the buffer when the buffer is full and the consumer won’t try to remove data from the buffer if the buffer is empty. For solving this concurrency problem, the producer and the consumer will have to communicate with each other. If the buffer is full, the producer will go to sleep and will wait to be notified. After the consumer will remove some data from the buffer, it will notify the producer, and then, the producer will start refilling the buffer again. The same process will happen if the buffer is empty, but in this case, the consumer will wait to be notified by the producer.

If this communication is not done properly, it can lead to a deadlock where both processes will wait for each other.

1. Create a simple client-server application Java using sockets. Client should send a message to server and get a message back from the server.
2. Create a simple chat application where the server will give appropriate answer to the question asked by the client.
3. Create a server application in Java that can support multiple clients.