Problem Set 3

(10 points) Cohort Exercise 1:

Write a Java program such that given a semi-prime, outputs its prime factors.

Name Your Solution: FactorPrime.java

(10 points) Cohort Exercise 3:

Write a Java program which allows a fixed N processes to talk in a fixed order (assume that a process is serving as a server).

Name Your Solution: ChatClient.java and ChatServer.java

(10 points) Cohort Exercise 5:

Continue with cohort exercise 1. Assume that you are given four computers, one acting as a server and the three clients do the actual work of factoring and display the result as soon as it is ready. The server knows the semi-prime and assigns the job.

Name Your Solution: FactorPrimeClient.java, FactorPrimeServer.java

(10 points) Cohort Exercise 6:

Continue with cohort exercise 5, assuming this time that you don't know how many clients are there to help you. Use socket timeout to collect clients and then assign the jobs.

Name Your Solution: FactorPrimeClientMul.java, FactorPrimeServerMul.java

(15 points) Homework Question 1:

Implement a simple chat room. There should be one process which works as the server and potentially one or multiple processes as the clients. The server is always waiting for client connection and messages. In reality, the messages from the clients are to be displayed on a webpage (which can then be viewed by anyone). In this exercise, assume the messages are simply displayed on the server's console. Any client can join and talk any time. Assume that the keyboard input "\n" from a client signals that he/she is done talking (for now at least). Hint: each client may make a new connection each time.

Name Your Solution: ChatRoom.java and ChatRoomClient.java

(15 points) Homework Question 2:

Extend the program developed in Question 1 so that the server waits for clients at the beginning (hint: use timeout so that if there is no new connection within 10 seconds, then start the chat) and the clients are each given 5 seconds to talk in sequence (e.g., the same sequence they join the chat room).

Name Your Solution: ChatRoom2.java and ChatRoomClient2.java

(10 points) Homework Question 3:

Implement a simple file transfer protocol (FTP) for text files. At the client side, the client transfers a text file (one line at a time) to the server and expects an acknowledgment from the server. Only after receiving an acknowledgment from the server, the client transmits the next line. If the acknowledgment is not received within a timeout period (choose your own value depending on your network delay), the client retransmits the unit. The above process is repeated until all the contents of the file are transferred. The server runs in an infinite loop for multiple clients (one after another) and writes the received file contents into a file.

Name Your Solution: FileTransfer.java and FileTransferClient.java

(10 points) Homework Question 4:

Develop a tic-tac-toe game. Two users connect to a server through sockets and take turns to make a move. The server closes the connections after a winner is decided.

Name Your Solution: TicTacToe.java and TicTacToePlayer.java

(Bonus 10 points) Homework Question 5:

Develop an election vote casting application as follows: There are two candidates A and B contesting an election. There are five electorates (processes) and each electorate can cast their vote only once and for only one of the two candidates (A or B). The vote cast by an electorate is a character ‘A’ or ‘B’, sent as a multicast message to all the other electorates. The winner is the candidate who gets the maximum number of votes. After casting the vote and also receiving the vote messages from all other electorates, each electorate should be able to independently determine the winner and display it. Hint: be aware of the deadlocks and try solving the problem using some systematic method.