Frequently Asked Questions

- Can we assume that the first command sent by a buyer/seller will always be the *login* command?
  Yes. If the first message of the client (seller or buyer) is a different command, the auction server can return an error message.
- In my program, when a buyer is outbid on a given item, the auction server broadcasts this information to all the buyers in the system (e.g. "User Tom has been outbid on item 10"). Is this fine?
  No. In the example above, only Tom should receive the message "You have been outbid on item 10".
- What do you mean exactly by the following sentence: "To get full credit, the incoming messages should not be delayed while waiting for input from the user at the client side; they must be displayed as soon as they are received."
  There are some design issues that you will need to pay special attention to. Suppose that Buyer 1 is currently the highest bidder on item X. While the client program for Buyer 1 is waiting for input from the user, it may be the case that Buyer 2 places a higher bid on item X. When the auction server receives the bid of Buyer 2, it will need to send a message to Buyer 1 ("You have been outbid on item X"). This incoming message to Buyer 1 will need to be displayed immediately; that is, to wait until Buyer 1 types a command and only then to display any waiting message is unrealistic (Perhaps Buyer 1 will not enter a new command for a long time). Similarly, the auction server must "transmit" messages to the target clients without any delay.
- I am designing my system in such a way that each buyer connects to a separate server port. Is this fine?
  No. All the buyers should connect to the same server port. This is mentioned in the hand-out.
- Can the auction server have a priori information about the port numbers of the clients?
  No. This is mentioned in the hand-out.
- What is the maximum number of buyers that may be connected to the AS simultaneously in your test cases? This information will help me to decide on the size of some data structures.
  At most six. Considering also the seller, there will be at most seven concurrent connections to AS.
- How will the auction server know that all the bids for a given item X are submitted, so that it can end the auction on that item?
  The auction on a given item X will continue until the seller decides to sell it [to the current highest bidder] through an explicit "sell" command.
- Can a buyer send multiple bids for an item?
  A buyer can send multiple bids for an item. The auction server will simply consider the highest bid sent by any user when the seller issues the "sell" command.
- When a buyer places a higher bid on a given item X, I am broadcasting this information to all the buyers in the system. Is this fine?
  No. Only the previous highest bidder needs to receive the message "You have been outbid on item X". And only the new highest bidder should receive the message "You are now the highest bidder on item X".
- Should the AS reject an "add" command with a duplicate item number? For example, the seller first issues the command "add 6 Algorithms Book". While this item is still on auction, the seller issues the command "add 6 Mozart CD". How should the AS react?
  This is an error condition. The AS should reject such an "add" command with a proper error message.
- In UDP implementation, it is possible that the messages will be lost during transmission. Should we design our system to guarantee "reliability" on top of UDP?
  No, that is beyond the scope of this project. You don't have to incorporate reliable packet delivery guarantees for UDP.
- Will there be only one seller process in the system?
  Yes. If you prefer, you can implement a system with multiple sellers, but there will not be any bonus points for this. However, your implementation must allow multiple buyer processes to be active at the same time.
- I am assuming that the seller always uses the username "Seller". When a client connects to the server with another username, I am assuming that this is a buyer. Is this acceptable?
  Yes.

- My understanding is that whenever a client/buyer sends out a command to the auction server, it should always receive some type of a reply from the server; at least an error message or an acknowledgment. Is that correct?
  Yes -- and this reply must be displayed on the client's terminal/window without a delay.
- I will submit a single server program for both TCP and UDP; an on-line argument will determine whether it will work with TCP or UDP. Is this fine?
  No, you should submit separate server programs for TCP and UDP. The same applies to the client (buyer and seller) programs. As the hand-out states, we are expecting six source files from you (compressed as a single file).
- Do we have to deal with the cases where the IP address/port number of a process changes dynamically? For example, a buyer may log on, send some commands, then exit. When the same buyer returns later it may have a different IP address/port. Moreover, a buyer may not be on-line when his/her offer has been outbid; how are we going to deliver this information to the buyer?
  You do not have to deal with these scenarios in this project. You may have noticed that you don't have to implement a "logout" command.
- Can we use the code provided in the web tutorials and TA's web page?
  Of course, you can. There are certain standard rules and socket call sequences that must be followed when initiating server and client processes in every socket programming application (Such as "socket", "bind", "listen", "accept" sequence for TCP-based sockets). The best way to learn these is in fact to examine example programs provided in the tutorials. Your server and clients will require some additional design, but the use of socket calls will be fairly similar.
- Should we design a separate program for each buyer??
  No, you will prepare only one buyer program for TCP and one buyer program for UDP. When testing (say) the TCP client/server connections, the user/grader will open multiple windows, and in each of them, invoke separate instances of the same buyer program. In other words, you will have multiple buyer processes running in different windows, and all these processes will be invoked through the same buyer executable program. Similarly, the auction server and the seller can be invoked in two different windows. If you are developing your program remotely by connecting to GMU machines, then invoke multiple ssh/ remote login sessions on different windows: the auction server, the seller and and each buyer can run in a window of their own.
- I am reading socket programming tutorials for C. I noticed that there are two types of sockets (STREAM SOCKETS and DATAGRAMS); which one should we use?
  STREAM SOCKETS work with TCP and DATAGRAMS work with UDP.
- Is there any way to disable sockets that have been forgotten to be closed, after program termination?
  The operating system periodically "resets" sockets which are not currently in use, so it may take a couple of minutes before the socket you forgot to close is reset.
- Since all ports below 1024 are reserved and some above are unavailable, is there any suggestion about the ports that would be reliable to try?
  You can choose a random number between 1024 and 65535; if this port is already in use by another student's program, then the system will warn you; and you can try another one. In many cases you may need to select your server port numbers only once, because the number of available ports is way too large when compared to the number of potential "socket programming projects" at GMU.
- What is the maximum number of incoming connections for the server, in other words, what should be the second parameter of the "listen" socket call issued by the server process (in C)?
  Setting it to 10 would be fine for this project.
- When I abort a program, all the socket connections between the server and the clients fail; and the programs exit with an error message. Will you be testing this type of scenario?
  We will not explicitly kill one process and check if the system is still functioning correctly.
- Can we use RPC/RMI in this project?
  No. This is a socket programming project.