Project 4 Rubric

Total (300 pts)

ALL communication between the bank, auction houses, and agents must be done through sockets.

Standards (75 pts)

- Follows coding standards (30 pts)
- Follows submission guidelines (same as project 3) (15 pts)
- Has design documents for the overall structure (15 pts)
- Has design documents for each piece of the system (15 pts)

System (100 pts)

- Bank is the only static known address (unless you are doing the extra credit) (10 pts)
- Can cope with *n* number of agents/auction houses (50 pts)
- Agents/Auction houses can all run on the same server (10 pts)
- Agents/Auction houses can run on separate servers (30 pts)

Bank (25 pts)

- Correctly blocks funds (20 pts)
- Correctly unblocks funds (20 pts)
- Agents can query for their bank account status (10 pts)

Auction House (50 pts)

- Starts up with at least 3 items for sale (10 pts)
- Bids cannot be placed with insufficient funds (20 pts)
- Auctions timeout after given amount of time (20 pts)
- Items are transferred to agent after agent releases payment to auction house (25 pts)

Agent (50 pts)

- Can query bank for account information (10 pts)
- Can correctly place bids (10 pts)
- Can connect to any active auction house (10 pts)
- Notifies user when they win an auction (10 pts)
- Notifies user when they are outbid on an auction (20 pts)
- De-registers with the bank when shutdown (15 pts)

Extra Credit (100 pts)

- Utilize JavaFX for the agent GUI (25 pts, pts based on functionality/look of GUI)
- Instead of each auction house having a copy of all the possible items for sale instead utilize a separate sever that holds the data. Auction houses then need to query this server over the network for a random subset of items to sell when they startup. (50 pts)
- On this data server utilize a database application to store the data (SQLite, Postgres, etc) (25 pts)