Project 2:

Program Description: You have been tasked with creating an auction program. This program will allow auctioneers to list items and customers to bid on items. In this specific application we decide that auctioneers decide when an auction is complete. You can assume that this program is run at time. This means that all data outside of usernames, passwords, and auctions are deleted. This should be done using proper object-oriented programming (Objects/classes, Encapsulation, and inheritance). See application specifics for all information pertaining to each object. Use the skeleton code provided to create and finish the program. All methods that have a comment which states "TODO" need to be done.

Application Specifications:

This program should allow for 3 types of users. A customer, auctioneer, and admin. Each performs separate duties. A customer can place bids, add money to their account, and add auctions to their watch lists. An Auctioneer can create auctions and manage them(start/stop). An Admin can only create Auctioneer accounts. Finally, an auction consists of 9 things; the name of the item, the category of the item, The time it was posted, starting amount of the bid, current bid, highest Bidder, status of the auction (upcoming, inprogress, complete), and which auctioneer is managing it. All auctions should be written to a txt file and saved.

Check out tasks to see specifics.

Add skeleton code to Eclipse:

1. Click File->Import
2. In Import Menu General->Existing Projects into Workspace
3. In "Select root directory" Place the AuctionSkeleton folder in.
4. Click on select All
5. Press Finish Button

Tasks

Task 1:

Import the given jar file called account.jar. This jar file contains methods which allow for the accounts of users to be created and written to files. The two methods which will be called from it are `addAccount(String username, String password)` and `signIn(String username, String password)`. The `addAccount()` method will write the username and password to a txt file for the user type. The `signIn()` method will return `true` if a username-password pair exists inside of the txt file.

EXAMPLE:

```
Accounts customers = new Accounts("Customers");
customers.addAccount("midas001", "Pass1234");
```

Task 2:

Inside of the auctions package there are 2 classes; `AllAuctions` and `Auctions`.
Follow subtasks for methods and Attributes.

Subtask 1: The `Auctions` Class

1.1: This class should have 9 Attributes. (`name, category, postedTime, starting Amounts, currentBid, highestBidder, auctionStatus, auctioneer`). Add these to the class
1.2: Complete the Constructor
1.3: Create setters and getters methods to be able to access the class member fields.
Note: (`postedTime` should be in local date time format, can use the `java.time` package). Remember that we load previous auctions as well as create new auctions so you may need to add some conditions in the constructor.

Subtask 2: The `AllAuctions` Class

2.1: Complete the `setAllAuctions` method.
2.2: Create setters and getters methods to be able to access the class fields. In the setter load in previous auctions from a txt file.
2.3: Complete the print auctions from a passed in list of objects. It should be printed as follows:

```
|     |Status     |Item Name      |Category       |Current Bid|Top Bidder     |
-------------------------------------------------------------------------------
|0    |In Progress|Phone          |Electronic     |150.00     |Ryan           |
|1    |Upcoming   |Chair          |Furniture      |0.00       |               |
|2    |Upcoming   |Shirt          |Clothing       |0.00       |               |
|3    |Upcoming   |keyboard       |electronics    |0.00       |               |
```

Task 3:

Inside the user package there are 4 classes. The first class will be the parent class of the other three. The parent class is `User`, the 3 subclasses are `Auctioneer`, `Admin`, and `Customer`.
Follow subtasks for methods and Attributes.

Subtask 1: The `Users` Class

1.1: This class should have 2 Attributes. (`Username` and `Password`)
1.2: Create setters and getters methods

Subtask 2: The `Auctioneer` Class

2.1: Complete the `setAuctions` method. You need to loop through all auctions and add any which are created by the auctioneer to the auctions array list in the auctioneer object.
2.2: Complete the `writeToAuctions()` method. Every time an auction is created write it to `Auctions.txt` file. The string that should be written is of the form:

```
[obj.getItemName()+","+obj.getCategory()+","+obj.getS
tartingAmount()+","+obj.getCurrentBid()+","+obj.getHighestB
idder()+","+obj.getAuctionStatus()+","+obj.getAuctioneer()]
```

2.3: Complete the method which adds an auction to all auctions list as well as the auctions list that an auctioneer manages. When an auction is added the status is "upcoming".
2.4: Complete the `manualAddition()` method.
2.5: Complete the `start` and `end` auctions methods. When an auction is "In Progress" it can be bidded on. When an auction is ended it becomes "Completed"

Subtask 3: The `Admin` Class

3.1: Complete the method which creates an auctioneer. This should take in the input of a username and password and call the `addAccount` method from jar account added in Task 1.

Subtask 4: The `Customer` Class

4.1: This class should have 3 attributes; an account balance, List of auctions that are being bided on, and a List of auctions on a customer's watch list (auctions which a customer is interested in).
4.2: Create setters and getters methods
4.3: Complete the `generateBids()`. This will load the add any auction that has the customers username as the top bidder.
4.4: Complete the `addToAccountBalance()`. This asks how much the user wants to add to their account balance and then adds it to the account balance.
4.5: Complete the `addAfterFailedBid()`

Note: You will need to pass the parameters of the previous top bidders username, amount of the previous top bid, and the array list of all customers.

4.6: Complete the method `makeBid()` which will allow a customer to make bids. A customer cannot place a bid for a "completed" or "upcoming" bid. A user should also have enough account balance to make a bid.
4.7: Complete the to add and delete from watch list methods. You should also complete the method which asks for whether you want to add or delete an auction from the watch list, `controlWatchList()`.

Task 4:

Follow the subtasks for the utility package

Subtask 1: Import the jar file into the `Utilities` Class. Once imported uncomment the 3 private variables at the top of the file.

Subtask 2: Complete all 4 Menu methods, a main menu and one for each of the user type. The menus should look as follows in console. All functionality from the screenshots below should be in your menus. Keep in mind that only `mainMenu()` should be called in the main method. In the User menus you should call all other methods pertaining to that class. This means to save all user objects you need to store them into the array lists from subtask 1. User menus should take in a username and password.

Main menu:

```
1. Create an account
2. Sign In
```

Customers menu:

```
Hello Customer Ryan
Welcome to the menu, What do you want to do?
1. View all auctions    2. Filter Auctions
3. View Watch List      4. Add/Remove from Watch List
5. View your Bids       6. Make A Bid
7. View My Balance      8. Add to Balance
9. Sign Out
```

Admin menu:

```
Hello Administrator Admin
Would you like to create an auctioneer account? (Y or N)
```

Auctioneer menu:

```
Hello Auctioneer Ryan
Welcome to the menu, What do you want to do?
1. Create an auction    2. View your Auctions
3. Start an Auction     4. End an Auction
5. Sign Out
```

Subtask 3: Complete the `sign in` method. This method should ask for the users type and then for the username and password of the user signing in. It should use the imported method `signIn()` from the jar file to check if a pair of username and passwords exists. If they do exist, direct them to the menu of their usertype.

Subtask 4: Complete the `customerAccountCreation` method. This method should ask for the username and password for the customer account being created. It should use the imported jar file method from task one to create an account.

Subtask 5: Complete the `writeToAuctions` method. This method should delete all contents in the Auction.txt file and write the AllAuctions list to the txt.

Grading:

- Compilation (10 points)
- Style: Comments, Indentations, Simplicity of main (5 points)
- Follows OOP (10 points)
- No given methods left empty (10 points)
- methods only do one thing (5 points)
- Tasks
    - Task 1: Importing jar (5 points)
    - Task 2: Utilities Class 15
        - Menus are fully functional (5 points)
        - Can Create a customer account (5 points)
        - Can Sign in (5 points)
    - Task 3: Auctions 10
        - 2 Classes (Auctions and all Auctions) (5 points)
        - Print auctions (5 points
    - Task 4: Users 30
        - 4 Classes (5 points)
        - Admins can create Auctioneer users (5 points)
        - Auctioneers can create auctions (5 points)
        - Customers can make bids (5 points)
        - Customers can view their bids (5 points)
        - Customers can add and remove from watch list (5 points)

Total: 100 points