Submission report - Lab 4: Implementing a design based on inheritance

24292-Object Oriented Programming

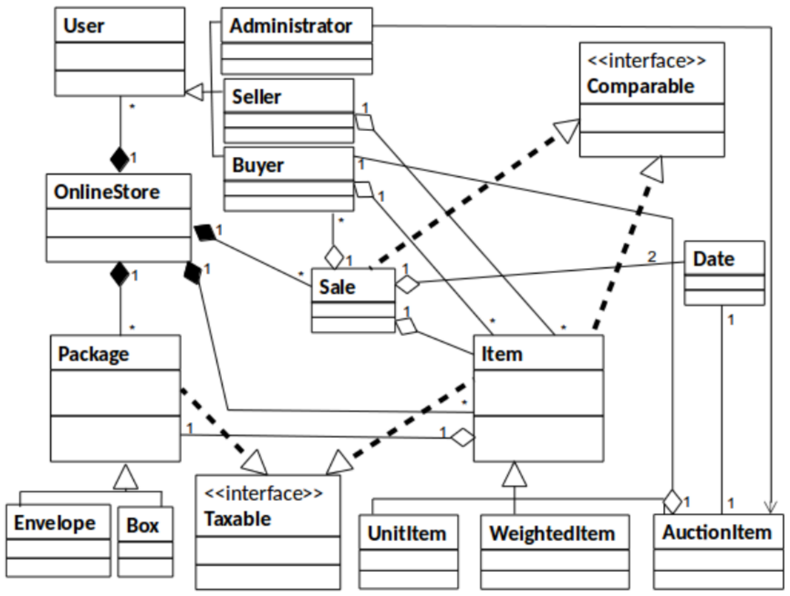
Name: Alexandur Surname: Orlinov NIA: 206312

Name: Ariela Surname: Strasser NIA:216893

****Same as the report of lab 3, with extended info on the changes.**

Introduction:

The aim of this lab session is to implement the design of Seminar 4 that partially models an online store. The project uses 3 super classes Item, User and Package. Each super class have its sub classes. Item has for subclasses Unit Item, Auction Item and Weighted Item. User has Administrator, Seller and Buyer. Package have Box and Envelope. It also has Class Online Store, its main class and there we execute the program. **It also has class Sale, and two interfaces Comparable and Taxable. Their purpose will be explained later.**



Item class:

In item class we define the items that we are going to sell in our online store. In this class we declare the item's name, type, size and cost. We also have method that calculated the best packaged based on the size of the item, which we want to buy.

Then we have the three types of items:
UnitItem: extension of Item but we add 2 more attributes unitPrice and quantity. This class have methods getPrice, CalculateProfit and Sell.

WeightedItem: extension of item with 2 more attributes PricePerWeight and Weight. Here we have the same methods as in UnitItem.

AuctionItem : extension of item with 3 additional attributes current price, Bidder and Deadline. Here we have the same methods as the previous two subclasses plus makeBid, Frozen , getBuyer and get Deadline Methods. **We have added current date, since we will need it in future method in Online Store.**

In class item we add the method Compare, which we will use later to compare items, in order to sort them by profit.

Also, we have added some methods, to calculate the price of un item with taxes included and without. Plus a function sumTotalTax that accumulate a total tax adding all taxes from taxable objects.

Item implements the interfaces taxable and Compare that's why we have added these functions inside the classes.

The same goes for Package class which implements taxable interface. We have added all the methods of Taxable. The purpose of this interface is to create a list of taxable items which later we will sort.

User Class:

In User call we define the type of users that we are going to have in our online store. In the class we declare the user's name, ID and password. We have getters and setters methods and login method. User class have 3 sub classes – Administrator, Buyer and Seller.

Administrator: extension of User class, we add list of users as attributes. Here we have method expel user, Manage Action and printStock. We also add linked list of users to the constructor of administrator since we have method expel user, and to be able to manipulate the list of users.

Seller: extension of User class, we add in attributes accountName, list of sold items, list of available items and deposit. As methods, we have sell item, AddAvailabeItem, and deposit and the constructor method. We also add linked lists of sold and available items to the constructor of seller so that the seller can manipulate the lists (AddAvailableItem and sell methods, need to update the lists of sold and available items)

Buyer: extension of User class, we add in attributes accountName, list of bought items and deposit. As methods, we have buy and pay, and the constructor method. We also add to the constructor the linked list of bought items, so that we can manipulate the list of bought items (with the function Buy)

Package class:

In Package class we define the type of packages that we are going to have in our online store. In the class we declare the package's width and height as attributes. We have getters and setters methods and the constructor method. We also have 2 subclasses Box and Envelope.

Box: extension of Package with extra attribute depth. Here we have constructor method and isSuitable method, which calculates if the item we want to pack is suitable for box.

Envelope: extension of Package with extra attribute name. Here we have constructor method and isSuitable method, which calculates if the item we want to pack is suitable for envelope.

MainOnlineStore : Lastly we have the class MainOnlineStore, where we have as attributes linked lists of available items, packages, users, sold items and bought items, totalPrice and total Profit with initial values 0.Here we add a function called Buy, which we use to buy items and when items are bought it calculates the total Profit and total Cost of the items We also have the Main Function in this class, where we execute our program. Inside the main we have added some sample items and users to check whether our methods work. **We have added two methods. CheckAuction and Sell methods, as required in the lab 4 document.**

Class Sale: We have created this class sale, with attributes Date, Item and buyer, in order to create a list of sold items where we can keep record of the item that was sold, the buyer and the date of the buy. This class implements the interface Compare. We do that because later, we will sort the sales by dates, using compareTo method.

Conclusion:

The theoretical concepts of object-oriented programming that we applied as part of the solution are: the application of fundamental concepts of classes, instances, methods and attributes, also visibility, the keyword this, and how to define relations between different classes, inheritance, polymorphism and abstract classes.

Overall, we managed to do everything without any problems. We used various websites like GeeksforGeeks or Stackoverflow to check methods and implementations of methods, how to use inheritance and etc. We also spent a lot of time testing weather everything was working. Something that occurred as an obstacle was that we did not know how to make functions like “sell” in the subclass Seller, because at first, we created in the constructor linked lists soldItems and availableItems as = new lists, which was our mistake. It took us some time to understand that we had to actually declare the list in main and afterwards put it as parameter in the constructor of sell. With that we fixed the other classes and methods, and the program finally started without problems.

The only thing that was an obstacle for us during this lab, was the sort methods. It took us time to understand how to implement and use the sort methods. We used GeeksForGeeks as study tool, and some videos in youtube.