Object Orientated Principals - Create an inventory/shopping menu-driven application as outlined below. Test program thoroughly so that it can handle incorrect user input (i.e. validate all user input, rather than just using exceptions) The program should keep looping until the user chooses to Quit.

PART 1

Create 8 java classes:

1. - A Product class.

This will be a generic class for Products that a company may sell. (consider inheritance here) It will have attributes for the following: name, description, price and productID. Use appropriate data types. Write getters and setters for all attributes. Write a method called print() that prints the Product information to the screen.
A unique productID is assigned to the product when it is created. You can use a static int to achieve this.

2. - A Phone class.

It will Inherit from the Product class. It will have attributes for make (Apple, Motorola, Samsung, etc), model (iPhone 6, Moto X, Galaxy S5, etc), and storage space (in gigabytes). Write getters and setters and Overload the print() method that's inherited from the Product class. Make use of the super.print() call.

3. - A TV class.

It will Inherit from the Product class. It will have attributes for make, screen size, type (LCD, LED, Plasma) and whether or not it is 3D capable. Write getters and setters and overload the print() method.

4. - Write a ProductDB class

This manages an ArrayList of Products. The class provides methods to add, remove and find a product. The find method returns a product object with a matching productID.

5. - Write an OrderDetails class

This has a Product and quantity attributes. Provide a Print method.

6. - Write an Order class

which manages an ArrayList of OrderDetails objects. Provide a Print method. Provide an add method which takes a product object and a quantity. The add method creates an OrderDetails object and adds it to the ArrayList.

7. - Write a customer class

that has attributes name, address. A Customer object also has an ArrayList of Order objects.

8. - Write a Test class

This instantiates a number of phones and TVs and sets their attributes. Create some Customer objects. Create a ProductDB object called database. Add all your products to the object. Create some Order objects for a Customer and add the orders to that Customer object. See example code below (Note not all parameters are shown). This is not the complete test class. It just involves putting some items into a database. You will need to expand on this test class.

```
Phone p = new Phone(“Apple”, “iphone6”,  ..); // all parameters not shown
Phone p1 = new Phone(“Samsung”,”Galaxy s6”, ...);
Phone p2 = new Phone(“Apple”, “iphone5”,  ..);
Phone p3 = new Phone(“Samsung”,”Galaxy s5”, ...);
ProductDB database = new ProductDB();
database.add(p1);
database.add(p);
database.add(p2);
database.add(p3);
Customer Mary = new Customer(“Mary” ..);
Order o = new Order();
```

```
o.add(p,12); // ordered 12 iphone 6 products
o.add(p1,1); //ordered 1 Galaxy s6
Mary.addOrder(o);
Order o1 = new Order();
o1.add(p2,1); // ordered 1 iphone 5 products
o1.add(p3,5); //ordered 5 Galaxy s5 products
Mary.addOrder(o1);
```

PART 2

create a UML class diagram which details all the java classes used in your application.

PART 3

Provide a menu in your Test class which has the following options:

1. Create a new product (phone or TV).

2. Search for a product by supplying the productid.

3. Display all products in the database.

4. Create a new customer.

5. Allow a customer to order some products by supplying the productID and quantity for each product.

You will need to use your find method on the database object to return the product object for the productID that was inputted by the user.

6. Display all the orders that a customer has made and all the OrderDetails that are in each order.

7. Quit

Example run below of the menu system user input shown in bold (this is a simplified example run which does not show all options or input validation/error handling):

Please enter a menu option

1. Create a new product (phone or TV).

2. Search for a product by supplying the productid.

3. Display all products.

4. Create a new customer.

5. Order Products.

6. Display all orders.

7. Quit

5

Enter the customer's name

Joe

Enter a product id and a quantity. Enter -1 to finish

2 3

You ordered 3 iphone 6

Enter a product id and a quantity. Enter -1 to finish

3 2

You ordered 2 galaxy s6

Enter a product id and a quantity. Enter -1 to finish

-1

Please enter a menu option

1. Create a new product (phone or TV).

2. Search for a product by supplying the productid.

3. Display all products.

4. Create a new customer.

5. Order Products.

6. Display all orders.

7. Quit

5

Enter the customer's name (Note: or you could also list the customers here and identify them

with a numerical index to select which customer is ordering the products)

Joe

Enter a product id and a quantity. Enter -1 to finish

2 1

You ordered 1 iphone 6

Enter a product id and a quantity. Enter -1 to finish

3 1

You ordered 1 galaxy s6

Enter a product id and a quantity. Enter -1 to finish

-1

Please enter a menu option

1. Create a new product (phone or TV).

2. Search for a product by supplying the productid.

3. Display all products.

4. Create a new customer.

5. Order Products.

6. Display all orders.

7. Quit

6

Enter the customer's name (Note: or you could also list the customers here and identify

them with a numerical index to select which customer is ordering the products similar to option 6)

Joe

The Orders for Joe are as follows:

Order 1:

3 iphones6

2 galaxy s6

Order 2:

1 iphones6

1 galaxy s6

Please enter a menu option

1. Create a new product (phone or TV).

2. Search for a product by supplying the productid.

3. Display all products.

4. Create a new customer.

5. Order Products.

6. Display all orders.

7. Quit

7

Goodbye