

**GTU Department of Computer
Engineering**

CSE 222/505 – Spring 2021

Homework 1 Report

ŞEYMA NUR CANBAZ

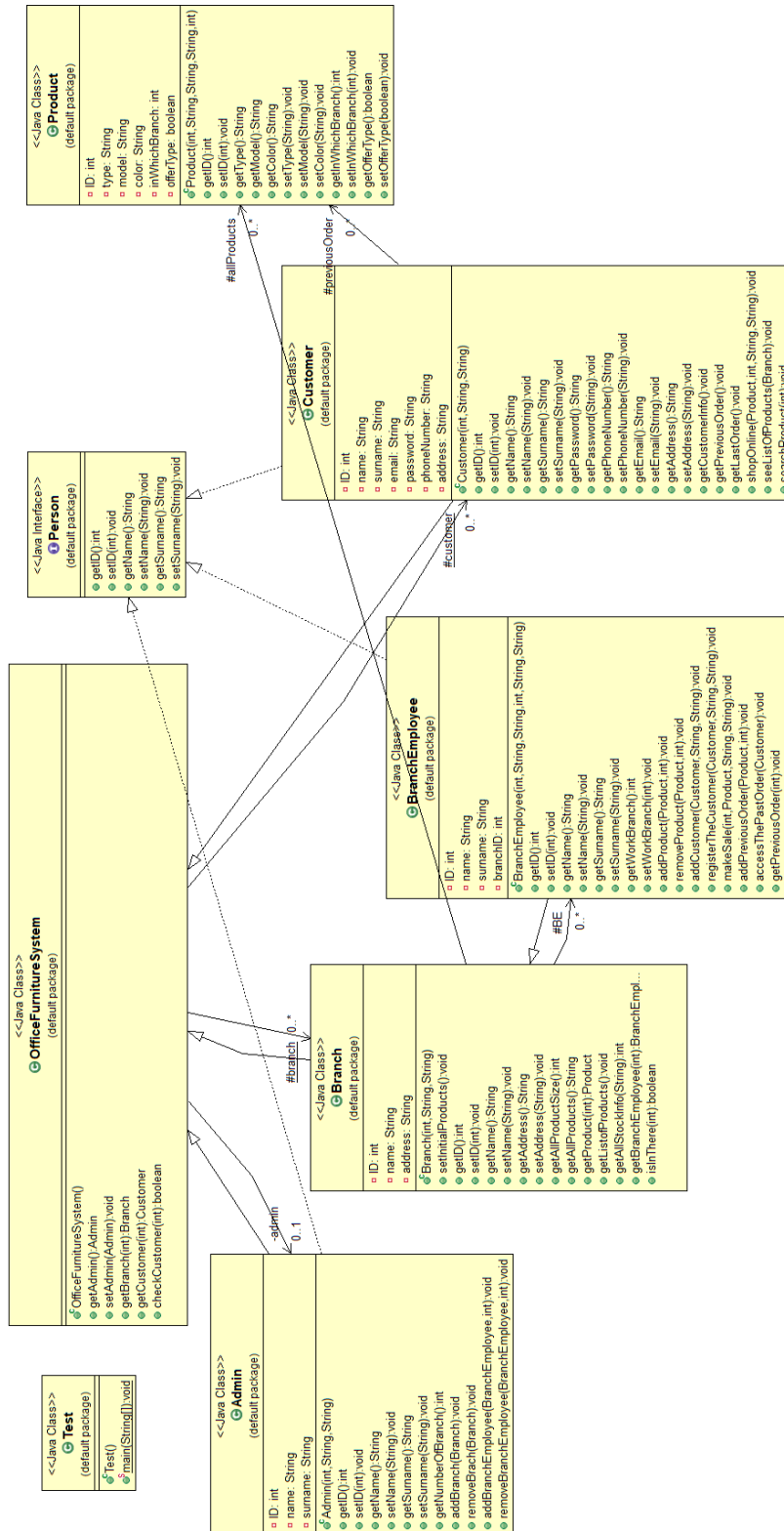
171044076

1. SYSTEM REQUIREMENTS:

In this assignment, we design an office furniture system. There is a manager managing the system, branch employees and customers using the system. Manager branches can add and remove employees. It can also query the stock of required products. Customers can view the list of products, search for products in the company, see which store the products are available in, shop online or in the store, and view their past orders. Customers must be a member of the system to be able to do all this. When they buy a product, they are asked for their address and telephone information. Branch employees can add and remove products, make sales in the store, access and update the customer's past orders, and create a record if the person who purchases a product from the store is not registered in the system.

For all this, the company needs a class. This class contains information about branches and customers. The manager manages these branches. Therefore, it should have an administrator class. In this class, there are some methods that the manager can add and remove his / her own information, branch and employee. Each branch has different products and employees. Therefore, there must be a branch class to keep this information. The branch employee queries the products in stock, adds and removes products to the system, creates customer orders and makes sales in the store. In order to keep both this information and the branch employee's own information, there must be a branch employee class derived from the branch class. To keep customers' own information and store their orders, it must be customer class. Therefore, there are many interconnected class structures.

2. CLASS DIAGRAMS:



3. PROBLEM SOLUTION APPROACH:

First, I created an "OfficeFurnitureSystem" class. Then I put the client array, the branch array, and a manager in this class. I needed branch class, manager class, and customer class for this data. First I created the Admin class. I have kept the ID, name and surname of the administrator for this class. I added functions to add and remove branches to the branch array in the "OfficeFurnitureSystem" class. I added methods for adding and removing employees to the BranchEmployee array in the branch class. Then I created the Branch class. I kept the branch ID, branch name, and address in this class. I created an array called allProducts to hold products within the branch. I wrote methods to query the product stocks at the branch. I have created a class named Product to keep product information. Here I kept the information about the product ID, type, model, color, which store it is located in, and if the sale was made, from where the sale was made. I also created a class called BranchEmployee to keep the BranchEmployee array inside the Branch class. In this class, I kept information such as the employee ID, name, surname, and the branch where he works. I have written methods so that the employee can query the product stocks, add and remove products, and make sales. At the same time, I created the addCustomer method to save the customer who bought the product and their past orders into the system. I created the Customer class to keep customers in the system. In this class, I created the previousOrder array to keep customers' past order information. I have added the necessary methods so that the customer can view the product information and place an order.

4. TEST CASES:

Test Case 1: Adding the admin in the company.

Test Case 2: Adding the branch in the company.

Test Case 3: Removing branch in the company.

Test Case 4: Adding the same branch in the company.

Test Case 5: Removing a branch that is not in the branch.

Test Case 6: Adding the branch employee in the branch.

Test Case 7: Removing the branch employee in the branch.

Test Case 8: Adding the same branch employee in the same branch again.

Test Case 9: Removing a branch employee that is not in the branch.

Test Case 10: Questioning the product stock with manager.

Test Case 11: Adding a new product in the branch.

Test Case 12: Removing the product in the branch.

Test Case 13: Adding the same product (have same ID) in the company.

Test Case 14: Removing a product that is not in the branch.

Test Case 15: Adding a customer in the company.

Test Case 16: Adding the same customer in the company.

Test Case 17: Seeing the list of product by customer.

Test Case 18: Search the product by customer.

Test Case 19: Search the product that is not exist by customer.

Test Case 20: View the customers previous order (by customer).

Test Case 21: View the customers previous order (by branch employee).

Test Case 22: Shopping online.

Test Case 23: Shopping in branch (with registered customer).

Test Case 24: Shopping in branch (not registered user).

Test Case 25: Updating customers' previous orders.

Test Case 26: Check the stock (in the branch where it is located)

Test Case 27: View the previous order by customer (if he/she has no orders).

Test Case 28: Check the product stock (using product id).

5. RUNNING COMMAND AND RESULTS:

Test Case 1: Adding the admin in the company. (Şeyma Canbaz 100000)

```
OfficeFurnitureSystem company = new OfficeFurnitureSystem();
Admin admin = new Admin(100000, "Şeyma", "CANBAZ");
company.setAdmin(admin);
System.out.println("Admin: " + company.getAdmin().getName() + " " + company.getAdmin().getSurname());
```

```
Admin: Şeyma CANBAZ
```

Test Case 2: Adding the branch in the company.

```
Branch b1 = new Branch(2000, "İstanbul", "İstanbul Street");
Branch b2 = new Branch(2001, "Ankara", "Ankara Street");
Branch b3 = new Branch(2002, "İzmir", "İzmir Street");
Branch b4 = new Branch(2003, "Antalya", "Antalya Street");

company.getAdmin().addBranch(b1);
System.out.println("Branch Name: " + company.getAdmin().getBranch(2000).getID() + "\n");

company.getAdmin().addBranch(b2);
System.out.println("Branch Name: " + company.getAdmin().getBranch(2001).getID() + "\n");

company.getAdmin().addBranch(b3);
System.out.println("Branch Name: " + company.getAdmin().getBranch(2002).getID() + "\n");

company.getAdmin().addBranch(b4);
System.out.println("Branch Name: " + company.getAdmin().getBranch(2003).getID() + "\n");
```

```
İstanbul is added
Branch ID: 2000

Ankara is added
Branch ID: 2001

İzmir is added
Branch ID: 2002

Antalya is added
Branch ID: 2003
```

Test Case 3: Removing branch in the company. (İzmir branch)

```
//REMOVE BRANCH
company.getAdmin().removeBranch(b3);
```

```
İzmir branch is removed
```

Test Case 4: Adding the same branch in the company. Throws exception. (İstanbul branch)

```
try {
    //ADD SAME BRANCH
    company.getAdmin().addBranch(b1); //throws exception
}
catch(Exception e){
    System.out.println(e);
}

java.util.NoSuchElementException: İstanbul branch is already added!
```

Test Case 5: Removing a branch that is not in the branch. Throws exception. (İzmir branch)

```

try {
    //REMOVE BRANCH (DOES NOT EXIST)
    company.getAdmin().removeBranch(b3); //throws exception
}
catch(Exception e){
    System.out.println(e);
}

```

[java.util.NoSuchElementException](#): There is no branch called İzmir

Test Case 6: Adding the branch employee in the branch.

```

BranchEmployee BE = new BranchEmployee(600000, "Ali", "CANBAZ", b1.getID(), b1.getName(), b1.getAddress());
BranchEmployee BE2 = new BranchEmployee(600001, "Gülşay", "CAN", b1.getID(), b1.getName(), b1.getAddress());

//ADD BRANCH EMPLOYEE
company.getAdmin().addBranchEmployee(BE, company.getBranch(2000).getID());
System.out.println("Branch ID: " + company.getBranch(2000).getBranchEmployee(600000).getWorkBranch());

company.getAdmin().addBranchEmployee(BE2, company.getBranch(2000).getID());
System.out.println("Branch ID: " + company.getBranch(2000).getBranchEmployee(600001).getWorkBranch());

```

```

Ali CANBAZ employee added in this branch
Branch ID: 2000
Gülşay CAN employee added in this branch
Branch ID: 2000

```

Test Case 7: Removing the branch employee in the branch.

```

//REMOVE BRANCH EMPLOYEE
company.getAdmin().removeBranchEmployee(BE, company.getBranch(2000).getID());

```

```

Ali CANBAZ employee is removed in this branch.

```

Test Case 8: Adding the same branch employee in the same branch again.

```

//ADD SAME EMPLOYEE IN THE SAME BRANCH
try {
    company.getAdmin().addBranchEmployee(BE2, company.getBranch(2000).getID());
}
catch(Exception e) {
    System.out.println(e);
}

```

[java.util.NoSuchElementException](#): Gülşay CAN employee is already added in this branch!

Test Case 9: Removing a branch employee that is not in the branch.

```

//REMOVE BRANCH EMPLOYEE (DOES NOT EXIST)
company.getAdmin().removeBranchEmployee(BE, company.getBranch(2000).getID());

```

```

There is no employee called Ali

```

Test Case 10: Questioning the product stock with admin. (in given branch ID)

```

//INQUIRE THE STOCK (BY ADMIN)
company.getAdmin().getBranch(2000).getListofProducts();

```

Type: officeDesk	Model: model3	Color: green	ProductID: 130017	Branch: Istanbul
Type: officeDesk	Model: model4	Color: green	ProductID: 130018	Branch: Istanbul
Type: officeDesk	Model: model5	Color: green	ProductID: 130019	Branch: Istanbul
Type: officeChair	Model: model1	Color: black	ProductID: 140008	Branch: Istanbul
Type: officeChair	Model: model2	Color: black	ProductID: 140009	Branch: Istanbul
Type: officeChair	Model: model3	Color: black	ProductID: 140002	Branch: Istanbul
Type: officeChair	Model: model4	Color: black	ProductID: 140003	Branch: Istanbul
Type: officeChair	Model: model5	Color: black	ProductID: 140004	Branch: Istanbul
Type: officeChair	Model: model6	Color: black	ProductID: 140005	Branch: Istanbul
Type: officeChair	Model: model7	Color: black	ProductID: 140006	Branch: Istanbul
Type: officeChair	Model: model11	Color: white	ProductID: 140007	Branch: Istanbul
Type: officeChair	Model: model2	Color: white	ProductID: 140008	Branch: Istanbul
Type: officeChair	Model: model3	Color: white	ProductID: 140009	Branch: Istanbul
Type: officeChair	Model: model4	Color: white	ProductID: 140010	Branch: Istanbul
Type: officeChair	Model: model5	Color: white	ProductID: 140011	Branch: Istanbul
Type: officeChair	Model: model6	Color: white	ProductID: 140012	Branch: Istanbul
Type: officeChair	Model: model7	Color: white	ProductID: 140013	Branch: Istanbul
Type: officeChair	Model: model11	Color: blue	ProductID: 140014	Branch: Istanbul
Type: officeChair	Model: model2	Color: blue	ProductID: 140015	Branch: Istanbul
Type: officeChair	Model: model3	Color: blue	ProductID: 140016	Branch: Istanbul
Type: officeChair	Model: model4	Color: blue	ProductID: 140017	Branch: Istanbul
Type: officeChair	Model: model5	Color: blue	ProductID: 140018	Branch: Istanbul
Type: officeChair	Model: model6	Color: blue	ProductID: 140019	Branch: Istanbul
Type: officeChair	Model: model7	Color: blue	ProductID: 140020	Branch: Istanbul
Type: officeChair	Model: model11	Color: green	ProductID: 140021	Branch: Istanbul
Type: officeChair	Model: model2	Color: green	ProductID: 140022	Branch: Istanbul
Type: officeChair	Model: model3	Color: green	ProductID: 140023	Branch: Istanbul
Type: officeChair	Model: model4	Color: green	ProductID: 140024	Branch: Istanbul
Type: officeChair	Model: model5	Color: green	ProductID: 140025	Branch: Istanbul
Type: officeChair	Model: model6	Color: green	ProductID: 140026	Branch: Istanbul
Type: officeChair	Model: model7	Color: green	ProductID: 140027	Branch: Istanbul
Type: officeChair	Model: model11	Color: purple	ProductID: 140028	Branch: Istanbul
Type: officeChair	Model: model2	Color: purple	ProductID: 140029	Branch: Istanbul
Type: officeChair	Model: model3	Color: purple	ProductID: 140030	Branch: Istanbul
Type: officeChair	Model: model4	Color: purple	ProductID: 140031	Branch: Istanbul
Type: officeChair	Model: model5	Color: purple	ProductID: 140032	Branch: Istanbul
Type: officeChair	Model: model6	Color: purple	ProductID: 140033	Branch: Istanbul

```
Product product1 = new Product(5555, "bookcase", "model2", "-", company.getAdmin().getBranch(2000).getID());
Product product2 = new Product(5565, "bookcase", "model2", "-", company.getAdmin().getBranch(2000).getID());
Product product3 = new Product(5575, "bookcase", "model2", "-", company.getAdmin().getBranch(2000).getID());
Product product4 = new Product(5275, "meetingTable", "model2", "-", company.getAdmin().getBranch(2000).getID());
Product product5 = new Product(5175, "officeChair", "model2", "-", company.getAdmin().getBranch(2000).getID());
Product product6 = new Product(6175, "officeChair", "model2", "-", company.getAdmin().getBranch(2000).getID());

//ADD PRODUCT
company.getAdmin().getBranch(2000).getBranchEmployee(600001).addProduct(product1, 2000);
company.getAdmin().getBranch(2000).getBranchEmployee(600001).addProduct(product2, 2000);
company.getAdmin().getBranch(2000).getBranchEmployee(600001).addProduct(product3, 2000);
```


(Show the product ID)

```
5555product is added!  
5565product is added!  
5575product is added!
```

Test Case 12: Removing the product in the branch.

```
//REMOVE PRODUCT  
company.getAdmin().getBranch(2000).getBranchEmployee(600001).removeProduct(product1, 2000);
```

```
5555 product is removed
```

Test Case 13: Adding the same product (have same ID) in the company.

```
//ADD PRODUCT(SAME ID)  
company.getAdmin().getBranch(2000).getBranchEmployee(600001).addProduct(product3, 2000);
```

```
5575product is already added. Please change the product ID
```

Test Case 14: Removing a product that is not in the branch. (controls the product ID)

```
//REMOVE PRODUCT (DOES NOT EXIST)  
company.getAdmin().getBranch(2000).getBranchEmployee(600001).removeProduct(product1, 2000);
```

```
There is no product called 5555
```

Test Case 15: Adding a customer in the company.

```
//ADD CUSTOMER  
company.getAdmin().getBranch(2000).getBranchEmployee(600001).addCustomer(customer1, "cansu@email.com", "12345");  
company.getAdmin().getBranch(2000).getBranchEmployee(600001).addCustomer(customer2, "zeynep@email.com", "zeynep");
```

```
Cansu Tan customer is added  
Email and password saved! Registration completed successfully  
Zeynep Sevim customer is added  
Email and password saved! Registration completed successfully
```

Test Case 16: Adding the same customer in the company. (with the same id)

```
//ADD CUSTOMER(SAME CUSTOMER ID)  
company.getAdmin().getBranch(2000).getBranchEmployee(600001).addCustomer(customer1, "cansu@email.com", "12345");
```

```
Cansu Tan customer is already added!
```

Test Case 17: Seeing the list of product by customer. (in given branch)

```
//SEE THE LIST OF PRODUCT (CUSTOMER)  
company.getCustomer(900001).seeListOfProducts(b1);
```

[illegible]

Test Case 18: Search the product by customer.

```
//SEARCH THE PRODUCT (CUSTOMER)
company.getCustomer(900001).searchProduct(5565);
System.out.println("\n");
```

Searching...

Product in Istanbul branch

Test Case 19: Search the product that is not exist by customer.

```
//CUSTOMER SEARCH THE PRODUCT (DOES NOT EXIST)
company.getCustomer(900001).searchProduct(6175);
System.out.println("\n");
```

```
Searching...
Product not found
```

Test Case 20: Shopping online.

```
//SHOPPING ONLINE
company.getCustomer(900001).shopOnline(product5, 600001, "Ceviz Street NO:15", "777 77 77");
```

```
The product added to the past sales list
5175 product is removed
Your order has been received!
```

```
PRODUCT ORDER INFORMATION
Product Order ID: 5175
Product Order Type: officeChair
Product Order Model: model2
Product Order Color: -
Product Order Status: online
Product Order in Branchİstanbul
```

```
CUSTOMER INFORMATION
ID: 900001
NAME: Zeynep
SURNAME: Sevim
ADDRESS: Ceviz Street NO:15
PHONE NUMBER: 777 77 77
```

Test Case 21: View the previous order (by customer).

```
//ACCESS THE PAST ORDER (BY CUSTOMER)
System.out.println("--YOUR PREVIOUS ORDERS--");
company.getCustomer(900001).getPreviousOrder();
System.out.println();
```

```
--YOUR PREVIOUS ORDERS--
```

```
1. PRODUCT
```

```
Product ID: 5175 Type: officeChair Model: model2 Color: - Status: online Order in Branch: İstanbul
```

Test Case 22: View the customers' previous order (by branch employee).

```
//ACCESS THE PAST ORDER (BY CUSTOMER)
System.out.println("--YOUR PREVIOUS ORDERS--");
company.getCustomer(900000).getPreviousOrder();
System.out.println();
```

```
1. PRODUCT
```

```
Product ID: 5175 Type: officeChair Model: model2 Color: - Status: online Order in Branch: İstanbul
```

Test Case 23: Shopping in branch (with registered customer).

```
//SHOP IN BRANCH (WITH REGISTERED CUSTOMER)
company.getAdmin().getBranch(2000).getBranchEmployee(600001).makeSale(900001, product6, "Yıldız Street", "374 47 24");
```

```
The product added to the past sales list
6175 product is removed
```

```
CUSTOMER INFORMATION
ID: 900001
NAME: Zeynep
SURNAME: Sevim
ADDRESS: Yıldız Street
PHONE NUMBER: 374 47 24
```

```
The product sold
```

Test Case 24: Shopping in branch (not registered user).

```
//SHOP IN BRANCH (WITH NOT REGISTERED CUSTOMER)
company.getAdmin().getBranch(2000).getBranchEmployee(600001).makeSale(9000, product6, "Yıldız Street", "374 47 24");
System.out.println("\n");
```

Customer not found! Please register

Test Case 25: Updating customers' previous orders.

```
public void makeSale(int customerID, Product orderProduct, String customerAddress, String customerPhone) {
    int flag = 0;

    for(Customer cs : customer) {
        if(cs.getID() == customerID) {
            cs.setAddress(customerAddress);
            cs.setPhoneNumber(customerPhone);
            addPreviousOrder(orderProduct, customerID);
            removeProduct(orderProduct, branchID);
            cs.getCustomerInfo();
            flag++;
            System.out.println("The product sold");
        }
    }
    if(flag == 0) {
        System.out.println("Customer not found! Please register");
    }
}

company.getAdmin().getBranch(2000).getBranchEmployee(600001).accessThePastOrder(customer2);

1. PRODUCT
Product ID: 5175 Type: officeChair Model: model2 Color: - Status: online Order in Branch: Istanbul
2. PRODUCT
Product ID: 6175 Type: officeChair Model: model2 Color: - Status: online Order in Branch: Istanbul
```

Test Case 26: Check the stock (in the branch where it is located).

```
//CONTROL THE STOCK
System.out.println("Office cabinet stock: " + company.getAdmin().getBranch(2000).getAllStockInfo("officeCabinet"));
System.out.println("Office chair stock: " + company.getAdmin().getBranch(2000).getAllStockInfo("officeChair"));
System.out.println("Office desk stock: " + company.getAdmin().getBranch(2000).getAllStockInfo("officeDesk"));
System.out.println("Bookcase stock: " + company.getAdmin().getBranch(2000).getAllStockInfo("bookcase"));
System.out.println("Meeting table stock: " + company.getAdmin().getBranch(2000).getAllStockInfo("meetingTable"));
System.out.println("\n");
```

Office cabinet stock: 12
Office chair stock: 36
Office desk stock: 20
Bookcase stock: 15
Meeting table stock: 41

Test Case 27: View the previous order by customer (if he/she has no orders).

```
//ACCESS THE PAST ORDER BY CUSTOMER (NO PREVIOUS ORDER)
System.out.println("--YOUR PREVIOUS ORDERS--");
company.getCustomer(900000).getPreviousOrder();
System.out.println();
```

--YOUR PREVIOUS ORDERS--
Order not found

Test Case 28: Check the product stock (using product id).

There is no product called 5175
There is no product called 6175