SWE 4301 - Object Oriented Concepts II

LAB 1: OOC Assessment

Task 1

Michael Scott is the manager of a big grocery store. He has to keep track of the inventory levels of different products. He also places requisition orders for different types of products in variable quantities, while their inventory level is below a threshold level. Since the number of different products in the grocery store is huge, mostly he could not track all products inventory level in due time.

Thus, he has asked for a software entity which can keep track of the inventory level of each product in his store. It should allow him to monitor the present inventory level and add new products in the store. He can also add a certain number of pieces to an existing product. However, it should also allow him to add as many products as he wishes. Thus, you have to use a dynamic array or list to add and store the product's information. Each type of product in his store has a unique identification number, product name, price, current inventory level, minimum allowable inventory level and usual requisition amount.

However, the software entity should allow Michael Scott to send a requisition request any time he wants. For the time being, we assume that the requisition request conducted by Manager Michael Scott is carried out immediately by adding the requisition amount to the current inventory level of the specified product if its current inventory level is less than minimum inventory level.

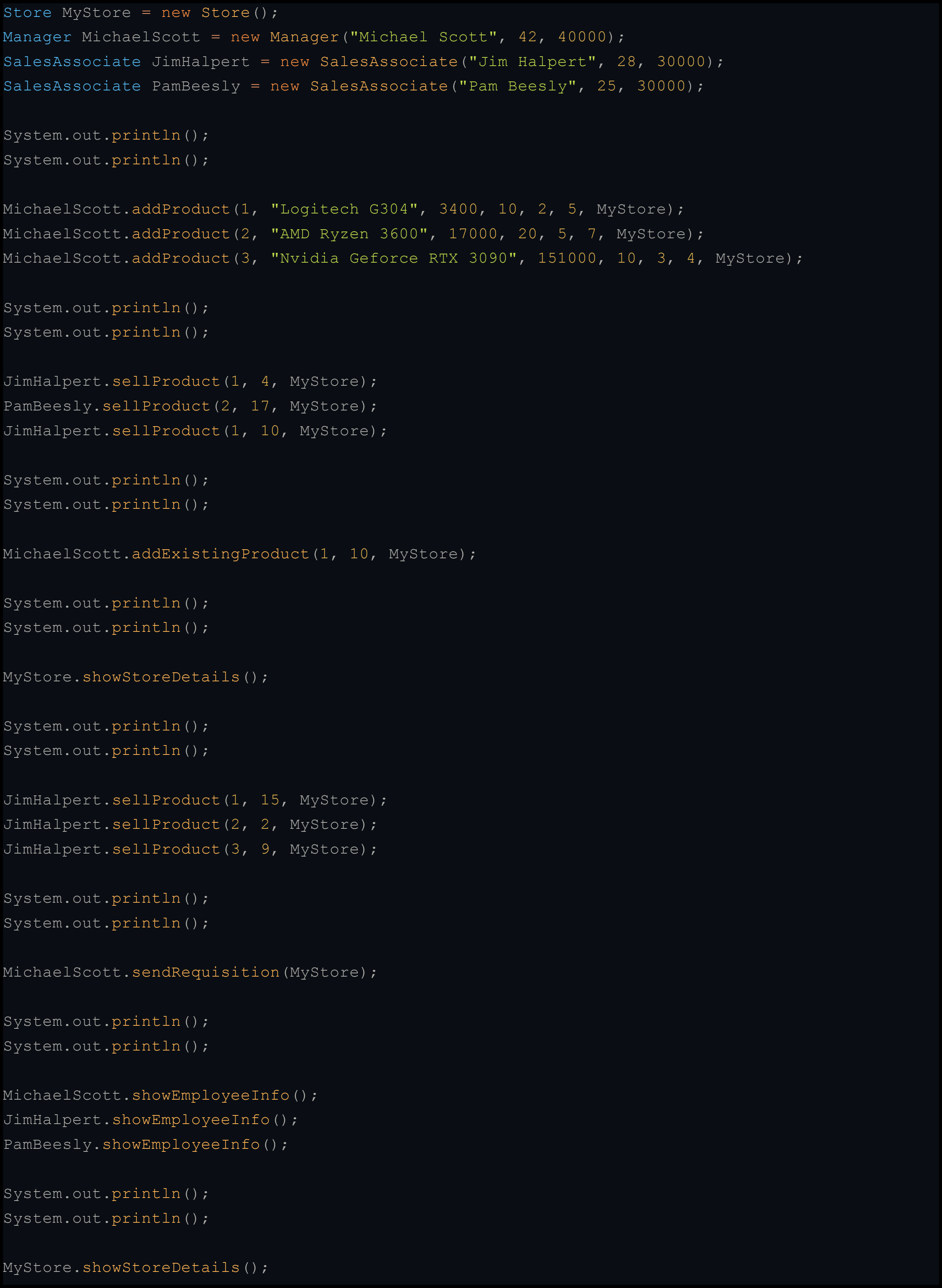
Besides, Michel Scott hires two new employees, Jim Halpert and Pam Beesly Both of them are sales associates. They are responsible for the sale of each item.

Now, let’s talk business, salary. Manager of a branch has a fixed salary. But a sales associate does not. S/he has a base salary. After every sale s/he makes, they get 1% of the total price of that sale as a bonus.

Now, by applying different object oriented concepts, define necessary classes with appropriate members to meet above mentioned requirements.

- You should use necessary abstract classes or methods to carry out the task.
- Make your program as modular as possible.
- Use meaningful class, variable and method names.

After completing the task, apply the following operations on your program.



```java
Store MyStore = new Store();
Manager MichaelScott = new Manager("Michael Scott", 42, 40000);
SalesAssociate JimHalpert = new SalesAssociate("Jim Halpert", 28, 30000);
SalesAssociate PamBeesly = new SalesAssociate("Pam Beesly", 25, 30000);

System.out.println();
System.out.println();

MichaelScott.addProduct(1, "Logitech G304", 3400, 10, 2, 5, MyStore);
MichaelScott.addProduct(2, "AMD Ryzen 3600", 17000, 20, 5, 7, MyStore);
MichaelScott.addProduct(3, "Nvidia Geforce RTX 3090", 151000, 10, 3, 4, MyStore);

System.out.println();
System.out.println();

JimHalpert.sellProduct(1, 4, MyStore);
PamBeesly.sellProduct(2, 17, MyStore);
JimHalpert.sellProduct(1, 10, MyStore);

System.out.println();
System.out.println();

MichaelScott.addExistingProduct(1, 10, MyStore);

System.out.println();
System.out.println();

MyStore.showStoreDetails();

System.out.println();
System.out.println();

JimHalpert.sellProduct(1, 15, MyStore);
JimHalpert.sellProduct(2, 2, MyStore);
JimHalpert.sellProduct(3, 9, MyStore);

System.out.println();
System.out.println();

MichaelScott.sendRequisition(MyStore);

System.out.println();
System.out.println();

MichaelScott.showEmployeeInfo();
JimHalpert.showEmployeeInfo();
PamBeesly.showEmployeeInfo();

System.out.println();
System.out.println();

MyStore.showStoreDetails();
```

The response/console output should look like this -

```
----Store has been generated successfully!----
----Manager has been added!----
Name: Michael Scott -- Age: 42 -- Base Salary: BDT40000
----Sales Associate has been added!----
Name: Jim Halpert -- Age: 28 -- Base Salary: BDT 30000 -- Bonus: BDT 0.0 -- Total Salary: BDT 30000.0
----Sales Associate has been added!----
Name: Pam Beesly -- Age: 25 -- Base Salary: BDT 30000 -- Bonus: BDT 0.0 -- Total Salary: BDT 30000.0


----Product has been added!----
ID: 1 -- Product Name: Logitech G304 -- Price: BDT 3400 -- Qty: 10 -- Min.Qty: 2 -- Req. Amt.: 5
----Product has been added!----
ID: 2 -- Product Name: AMD Ryzen 3600 -- Price: BDT 17000 -- Qty: 20 -- Min.Qty: 5 -- Req. Amt.: 7
----Product has been added!----
ID: 3 -- Product Name: Nvidia Geforce RTX 3090 -- Price: BDT 151000 -- Qty: 10 -- Min.Qty: 3 -- Req. Amt.: 4


Logitech G304 has been sold and Jim Halpert has received a bonus of BDT 136.0!
AMD Ryzen 3600 has been sold and Pam Beesly has received a bonus of BDT 2890.0!
Not enough amount of Logitech G304. Contact your manager!


----Added 10 pieces of Logitech G304----


----Inventory Details----
ID: 1 -- Product Name: Logitech G304 -- Price: BDT 3400 -- Qty: 16 -- Min.Qty: 2 -- Req. Amt.: 5
ID: 2 -- Product Name: AMD Ryzen 3600 -- Price: BDT 17000 -- Qty: 3 -- Min.Qty: 5 -- Req. Amt.: 7
ID: 3 -- Product Name: Nvidia Geforce RTX 3090 -- Price: BDT 151000 -- Qty: 10 -- Min.Qty: 3 -- Req. Amt.: 4


Logitech G304 has been sold and Jim Halpert has received a bonus of BDT 510.0!
AMD Ryzen 3600 has been sold and Jim Halpert has received a bonus of BDT 340.0!
Nvidia Geforce RTX 3090 has been sold and Jim Halpert has received a bonus of BDT 13590.0!


----Added 5 pieces of Logitech G304----
----Added 7 pieces of AMD Ryzen 3600----
----Added 4 pieces of Nvidia Geforce RTX 3090----


Name: Michael Scott -- Age: 42 -- Base Salary: BDT40000
Name: Jim Halpert -- Age: 28 -- Base Salary: BDT 30000 -- Bonus: BDT 14576.0 -- Total Salary: BDT 44576.0
Name: Pam Beesly -- Age: 25 -- Base Salary: BDT 30000 -- Bonus: BDT 2890.0 -- Total Salary: BDT 32890.0


----Inventory Details----
ID: 1 -- Product Name: Logitech G304 -- Price: BDT 3400 -- Qty: 6 -- Min.Qty: 2 -- Req. Amt.: 5
ID: 2 -- Product Name: AMD Ryzen 3600 -- Price: BDT 17000 -- Qty: 8 -- Min.Qty: 5 -- Req. Amt.: 7
ID: 3 -- Product Name: Nvidia Geforce RTX 3090 -- Price: BDT 151000 -- Qty: 5 -- Min.Qty: 3 -- Req. Amt.: 4
```