Description:

Supermarket management system must allow *store managers* to manage the supermarket's inventory, sales, cashiers, customers, suppliers, and shipments.

Classes:

Manager

- Product
- Sale
- Cashier
- Customer
- Supplier
- Shipment

Functional Requirements:

The Supermarket Management System provides different types of services where the store manager is the main user; We assume that the store does not have self-checkout. We also assume that the manager adds information for products and shipments, and that information does not come from an outside source. We also assume that all suppliers will have unique names.

The functions that are available to the manager include:

- Storing and checking current date
- Viewing upcoming shipments
- Adding and receiving shipments
- Adding, Removing, and viewing information about customers
- Adding, Removing, and viewing information about cashiers
- Generating a low stock report for products
    - Get a list of items that are lower than a percentage stored in an instance of the product itself
- Searching inventory for a product
- Displaing entire inventory
- Adding and removing products
- Seeing the most sold item in a specific time frame
- Seeing the number of sales made in a specific timeframe
- Seeing all sales made
- Adding, Removing, and Modifying information about suppliers

- Search for specific suppliers
- See all suppliers

Classes

*Implicit Setters and Getters for every Class

Manager/Main Class

int currentMonth //the month (mm format (01 is january))

int currentDay //the day (dd format 01 is first of the month)

int currentYear //the year (yyyy format)

String latestLowStockReport //most recently generated lowStockReport

String latestLowStockReportDate //date of most recently generated lowStockReport

HashMap<Integer, Cashier> cashiers //Key is id, must be unique

HashMap<Integer, Supplier> suppliers //Key is id, must be unique

HashMap <productName: String, product: Product> inventory //because product names should be unique

ArrayList <Integer> inventoryPriceOrder //indices of products in inventory arranged in ascending order by price, used for ease of access in displayInventory() method

ArrayList <Integer> inventoryStockOrder //indices of products in inventory arranged in ascending order by stock percentage, used for ease of access in displayInventory() method

ArrayList<String> inventoryAlphabetOrder //Keys for alphabetical order

HashMap<Integer, Customer> customers //Key is id, must be unique

ArrayList<Sale> sales //it'd be good to make sure sales are added in reverse chronological order to save time on searches

---

Manager(currentMonth: int, currentDay: int, currentYear: int) //variables to initialize on program startup, prompt user for the current date

void incrementDay() //Updates currentMonth, currentDay, and currentYear to represent the next day as per the calendar.

String currentDayToStr() //Returns the date in form mm/dd/yyyy

String generateLowStockReport () //creates a string containing the names of all products that are low supply //also updates latestLowStockReport variable //prints success or failure of operation

void displayLowStockReport() //Displays the latest low stock report to the terminal

void addShipment (int id, int arrivalMonth, int arrivalDay, int arrivalYear, String supplierName) //Creates and adds and instance of shipment to stored in the associated supplier

void removeShipment (int id, String supplierName) //Removes a specified shipment

void updateShipment(int id, int arrivalMonth, int arrivalDay, int arrivalYear, String supplierName) //Updates shipment information

String getUpcomingShipments(timeframe: String) //returns a table where each column is the supplier, and the rows after are the shipments with their information //timeframe will either be “day” (for shipments of the day), “month” (for shipments in the current month), or “year” (for shipments across the year).

String getUpcomingShipments() //without a parameter this function returns all upcoming shipments in the same format as above

String receiveShipment(supplier: Supplier, id: int) //removes the shipment of matching id from the supplier’s upcomingShipments and updates inventory accordingly. //returns a String informing whether the operation was successful or not and any errors (such as overcapacity on a product or shipment id number not existing).

String peekShipment(String supplierName, int id) //Get information about an upcoming shipment

Product searchInventory(productName: String) //searches inventory and returns Product object that shares a name with productName argument

void displayInventory (criteria: int, toFile: boolean) //prints the inventory arranged in different ways depending on the criteria argument, 0 for chronological, 1 for alphabetical, 2 for price descending, 3 for price ascending, 4 for percentage stock descending, and 5 for percentage stock ascending. //if toFile is true, will instead create a file with the method output rather than print to screen. A message will be printed to inform the user when the file is successfully created. This variation creates files with a naming schema differing by date and will overwrite a file if it shares an exact date with the one being created.

String mostSoldItem(timeframe: String) //returns the name of the most sold item within the time period passed, either “day”, “month”, or “year”

int numSold(productName: Product, timeframe: String) //returns the number of times the specified item has been sold within the timeframe (either “day”, “month”, or “year”)

int numSales(timeframe: String) //returns the number of sales that occurred within specified timeframe (either “day”, “month”, or “year”)

double averageSalePrice(timeframe: String) //gets the average total cost of all sales within the specified timeframe (either “day”, “month”, or “year”)

void addProduct(productName: String, cost: double, currentStock: int, maxStock: int, lowPercentage: int) //creates a new product object and adds it to Hashmap inventory (product names must be unique, will update product instead by behavior of hashmap, somehow indicate the difference to user

void removeProduct(productName: String) //removes product from inventory

//basic cashier management methods

void addCashier(firstName: String, lastName: String, id: int, job: String, salary: int) //creates a new cashier and adds to corresponding arrayList

void removeCashier(id: int) //removes cashier by id number

ArrayList<Cashier> searchCashierByName (String first, String last) //returns an array list of cashier objects that matches first and last name

ArrayList<Cashier> searchCashierByName (String first) //returns a list of all cashiers with matching first name

Cashier searchCashierById (int id) //Returns an object cashier with matching id

void displayCashiers() //prints out a list of all cashiers and their information

//basic customer management methods

void addCustomer(firstName: String, lastName: String, rewardsNumber: int, phoneNumber: String) //creates a new customer and adds it to corresponding arrayList //purchasesMade attribute is set to 0 by default

void removeCustomer(rewardsNumber: int) //removes customer based on rewards number (should be unique for each customer)

ArrayList<Customer> searchCustomerByName (String first, String last) //returns an array list of customer objects that matches first and last name

ArrayList<Customer> searchCustomerByName (String first) //returns a list of all customers with matching first name

void displayCustomers() //prints out a list of all customers and their information

//basic supplier management methods

void addSupplier(String name) //Creates and adds an instance of Supplier to suppliers

void modifySupplier (String oldName, String newName) //Updates the name of a supplier

Supplier searchSupplierByName(String name) //Returns a supplier by name

void displaySuppliers() //Displays all suppliers

//basic sale management methods

void addSale(dayOfSale: int, monthOfSale: int, yearOfSale: int, managingSale: Cashier, makingSale: Customer) //creates new sale object and adds it to the front (or 0^{th} index) of Sales arrayList

void addSale(dayOfSale: int, monthOfSale: int, yearOfSale: int, managingSale: Cashier) //creates new sale object and adds it to the front (or 0^{th} index) of Sales arrayList, this one does not require a customer to be passed

void removeSale(index: int) //removes sale object by its index in arrayList

void displaySales () //displays all sales with their information

boolean sellItem(String productName, int numSold) //Returns information about whether a product is sold

void removeSale(int id) //Removes an instance of sale

void displaySales() //Displays all sales that have been made

String getLatestLowStockReport() //Returns the latest low stock report that has been generated

void main () //Displays menuing that allows access to all the features avaliable to manager

Supplier

String name //name of supplier

private HashMap<String, Product> products //names of products delivered by the supplier, and instances of those products

HashMap<id: Integer, shipment: Shipment> upcomingShipments

Supplier (name, products) //constructor

String[] getLowProducts (masterReport: String) //parses the total low stock report from the generateLowStockReport() method in manager class, returns an array of all product names that are supplied by this supplier (defined by String[] products attribute)

void addShipment(id: int, arrivalMonth: int, arrivalDay: int, arrivalYear: int) //creates a new shipment and adds it to upcomingShipments

void updateShipment(id: int, arrivalMonth: int, arrivalDay: int, arrivalYear: int) //updates shipment with corresponding id number

void removeShipment(id: int) //removes a shipment by its id, does nothing if it doesn't exist

Shipment getShipment(int id) //Returns a shipment by a specified ID

HashMap<Integer, Shipment> getAllShipments() //Returns all shipments with their ID numbers

void addProduct(String name, double cost, int currentStock, int maxStock, int lowPercentage) //Creates and adds an instance of product to products

void addProduct(Product product) //Adds an instance of product that has already been created

void removeProduct(String productName) //Removes a product by name

Product

String name

double cost

int currentStock //current amount of this item on hand

int maxStock //total amount of this item the store can hold

int lowPercentage //threshold for defining the product as "low" as percentage

---

Product (name: String, cost: int, currentStock: int, maxStock: int) //default low percentage is 20%

Product (name: String, cost: int, currentStock: int, maxStock: int, lowPercentage: int)

int getStockPercentage () //returns the current stock level as a percentage //floors percentage if not whole

double increasePrice (percentage: double) //increases the price based on percentage passed, returns new price

double decreasePrice (percentage: double) //decreases the price based on percentage passed, returns new price

Shipment

int idNumber //an id number for the shipment

int arrivalMonth //month of arrival

Int arrivalDay //day of arrival

Int arrivalYear //year of arrival

HashMap<productName: String, product: Product> items //name of products coming in and how many

---

Shipment(id: int, arrivalMonth: int, arrivalDay: int, arrivalYear: int) //productList is empty

String arrivalDateToStr() //returns the date of arrival as mm/dd/yyyy for reporting purposes

void addProduct(name: String, cost: double, currentStock: int, maxStock: int, lowPercentage: int) //adds a new product to product list, note that product names must be unique, otherwise the product will be updated instead. //also the product name and the name it is mapped to in the hash map must match as well. //if the product does not exist in inventory yet, ask the user to define a maxStock and lowPercentage for the product, otherwise use the values of the product that is already there.

void removeProduct(productName: String) //removes item from list

Product searchProduct(String productName) //Returns a product in the shipment

int getProductCount(String productName) //Returns the stock of a specified product by name

String getProductInfo() //Returns a string containing the name and stock of every item in the shipment

Cashier

String firstName

String lastName

int id

int salary

Cashier(id: int, firstName: String, lastName: String, job: String, salary: int)

void increaseSalary (percent: double) //increases the salary of the cashier by a percentage

Customer

String firstName

String lastName

int customerId

String phoneNumber

int purchasesMade

Customer(firstName: String, lastName: String, customerId: int, String: phoneNumber)

void incrementPurchase() //increments purchasesMade variable by +1

Sale

int id

int dayOfSale

int monthOfSale

int yearOfSale

Cashier managingSale

HashMap <Product, Integer> productsSold //Contains the product sold as the key, and the amount of the product sold

Customer makingSale

Sale(managingSale: Cashier, productsSold: ArrayList <Product>) //Constructor is if the Customer is not in the system

Sale(managingSale: Cashier, productsSold: ArrayList <Product>, makingSale: Customer) //Constructor if customer is in the system

int numberOfProductsSold() //Returns how many items were involved in the transaction

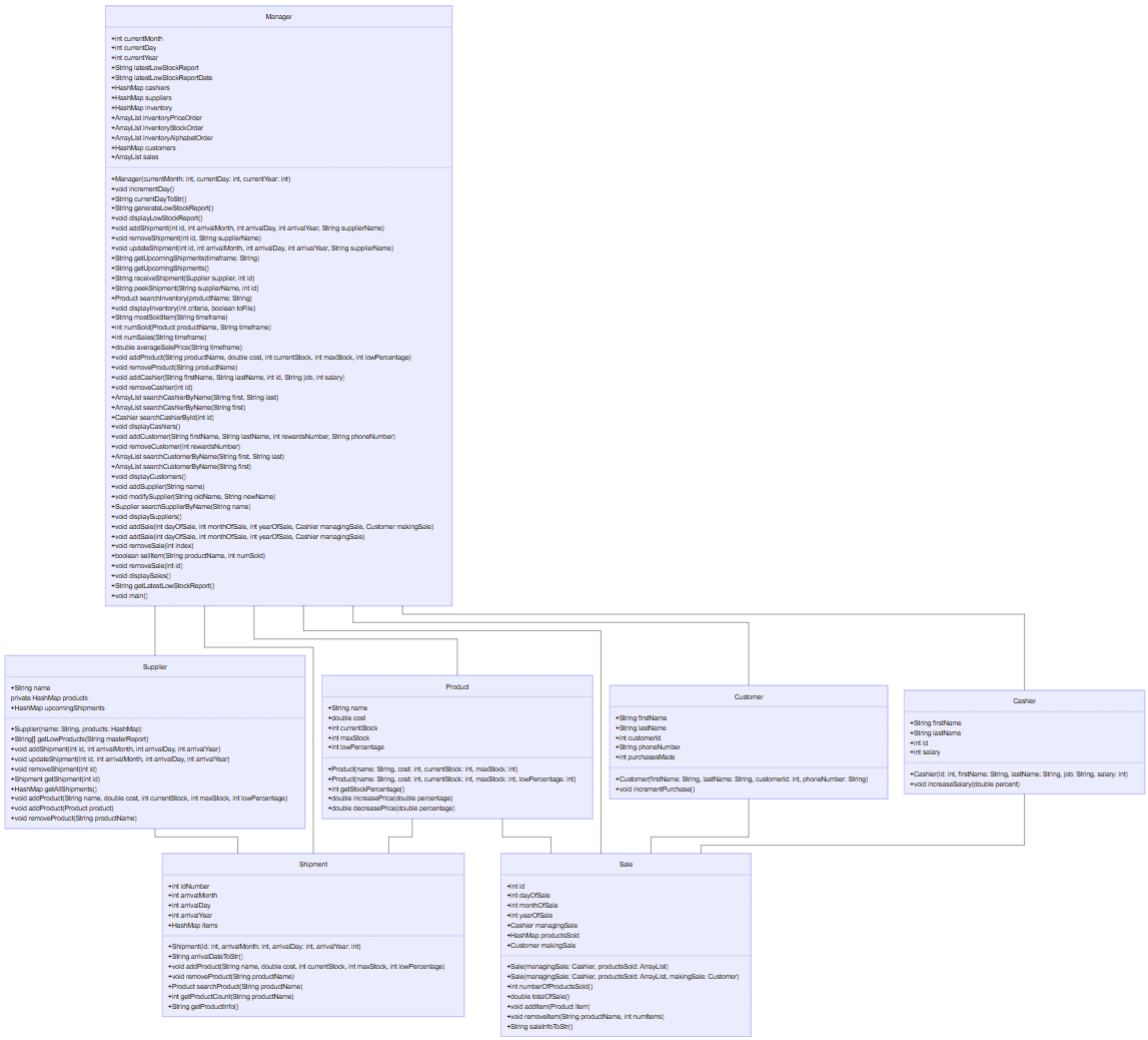
double totalOfSale() //Returns the total of the transaction

void addItem(item: Product) //adds passed product to productsSold arraylist

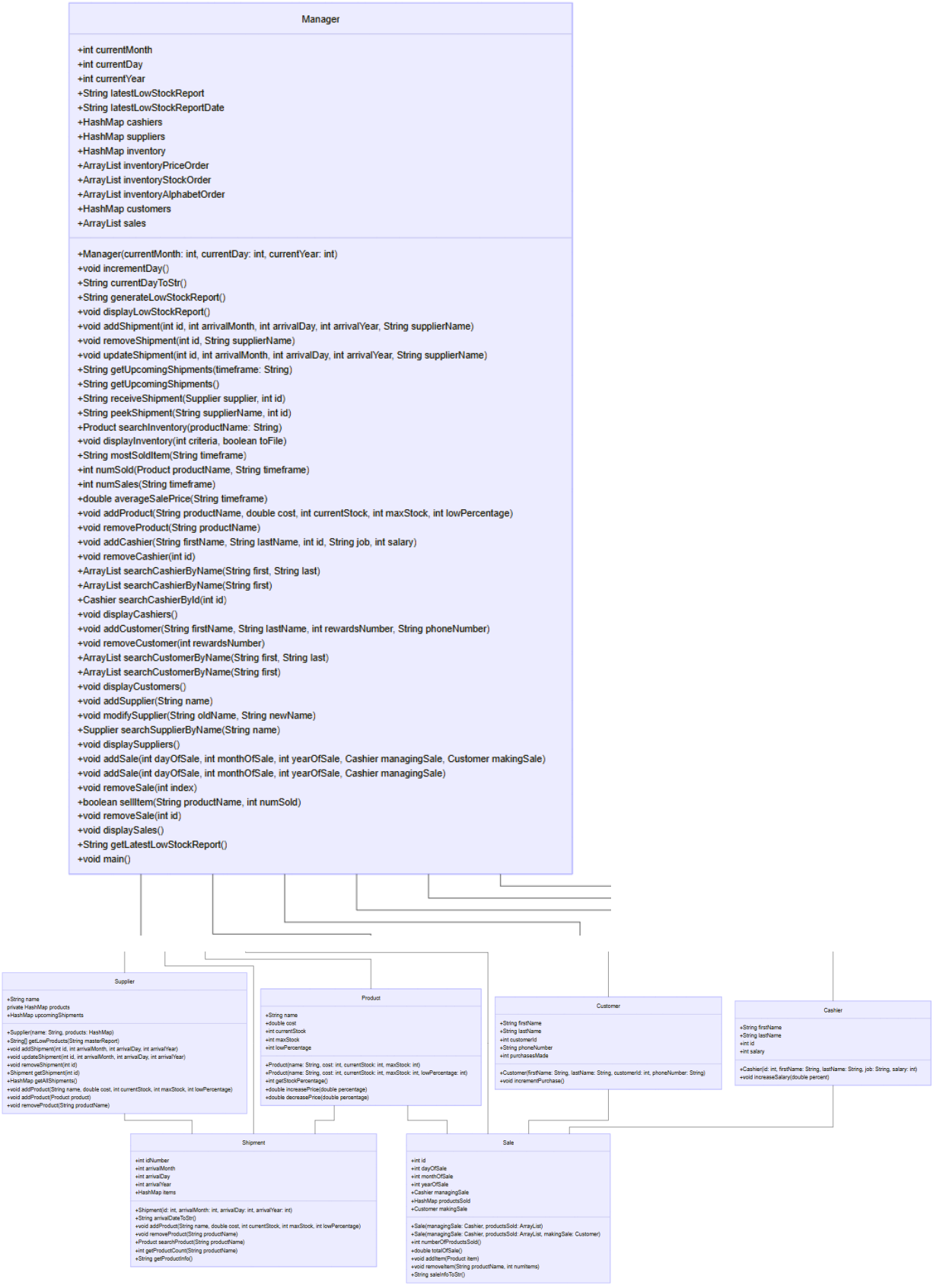
void removeItem(String productName, int numItems) //Decrements the amount of that product sold in map by numItems, completely removes the item is the amount falls below 0

String saleInfoToStr() //Returns a string containing all the information about the sale

Class Diagram



*Closeup on Manager and other classes


Manager
+int currentMonth
+int currentDay
+int currentYear
+String latestLowStockReport
+String latestLowStockReportDate
+HashMap cashiers
+HashMap suppliers
+HashMap inventory
+ArrayList inventoryPriceOrder
+ArrayList inventoryStockOrder
+ArrayList inventoryAlphabetOrder
+HashMap customers
+ArrayList sales
+Manager(currentMonth: int, currentDay: int, currentYear: int)
+void incrementDay()
+String currentDayToStr()
+String generateLowStockReport()
+void displayLowStockReport()
+void addShipment(int id, int arrivalMonth, int arrivalDay, int arrivalYear, String supplierName)
+void removeShipment(int id, String supplierName)
+void updateShipment(int id, int arrivalMonth, int arrivalDay, int arrivalYear, String supplierName)
+String getUpcomingShipments(timeframe: String)
+String getUpcomingShipments()
+String receiveShipment(Supplier supplier, int id)
+String peekShipment(String supplierName, int id)
+Product searchInventory(productName: String)
+void displayInventory(int criteria, boolean toFile)
+String mostSoldItem(String timeframe)
+int numSold(Product productName, String timeframe)
+int numSales(String timeframe)
+double averageSalePrice(String timeframe)
+void addProduct(String productName, double cost, int currentStock, int maxStock, int lowPercentage)
+void removeProduct(String productName)
+void addCashier(String firstName, String lastName, int id, String job, int salary)
+void removeCashier(int id)
+ArrayList searchCashierByName(String first, String last)
+ArrayList searchCashierByName(String first)
+Cashier searchCashierById(int id)
+void displayCashiers()
+void addCustomer(String firstName, String lastName, int rewardsNumber, String phoneNumber)
+void removeCustomer(int rewardsNumber)
+ArrayList searchCustomerByName(String first, String last)
+ArrayList searchCustomerByName(String first)
+void displayCustomers()
+void addSupplier(String name)
+void modifySupplier(String oldName, String newName)
+Supplier searchSupplierByName(String name)
+void displaySuppliers()
+void addSale(int dayOfSale, int monthOfSale, int yearOfSale, Cashier managingSale, Customer makingSale)
+void addSale(int dayOfSale, int monthOfSale, int yearOfSale, Cashier managingSale)
+void removeSale(int index)
+boolean sellItem(String productName, int numSold)
+void removeSale(int id)
+void displaySales()
+String getLatestLowStockReport()
+void main()
Supplier
+String name
private HashMap products
+HashMap upcomingShipments
+Supplier(name: String, products: HashMap)
+String[] getLowProducts(String masterReport)
+void addShipment(int id, int arrivalMonth, int arrivalDay, int arrivalYear)
+void updateShipment(int id, int arrivalMonth, int arrivalDay, int arrivalYear)
+void removeShipment(int id)
+Shipment getShipment(int id)
+HashMap getAllShipments()
+void addProduct(String name, double cost, int currentStock, int maxStock, int lowPercentage)
+void addProduct(Product product)
+void removeProduct(String productName)
Product
+String name
+double cost
+int currentStock
+int maxStock
+int lowPercentage
+Product(name: String, cost: int, currentStock: int, maxStock: int)
+Product(name: String, cost: int, currentStock: int, maxStock: int, lowPercentage: int)
+int getStockPercentage()
+double increasePrice(double percentage)
+double decreasePrice(double percentage)
Customer
+String firstName
+String lastName
+int customerId
+String phoneNumber
+int purchasesMade
+Customer(firstName: String, lastName: String, customerId: int, phoneNumber: String)
+void incrementPurchase()
Cashier
+String firstName
+String lastName
+int id
+int salary
+Cashier(id: int, firstName: String, lastName: String, job: String, salary: int)
+void increaseSalary(double percent)
Shipment
+int idNumber
+int arrivalMonth
+int arrivalDay
+int arrivalYear
+HashMap items
+Shipment(id: int, arrivalMonth: int, arrivalDay: int, arrivalYear: int)
+String arrivalDateToStr()
+void addProduct(String name, double cost, int currentStock, int maxStock, int lowPercentage)
+void removeProduct(String productName)
+Product searchProduct(String productName)
+int getProductCount(String productName)
+String getProductInfo()
Sale
+int id
+int dayOfSale
+int monthOfSale
+int yearOfSale
+Cashier managingSale
+HashMap productsSold
+Customer makingSale
+Sale(managingSale: Cashier, productsSold: ArrayList)
+Sale(managingSale: Cashier, productsSold: ArrayList, makingSale: Customer)
+int numberOfProductsSold()
+double totalOfSale()
+void addItem(Product item)
+void removeItem(String productName, int numItems)
+String saleInfoToStr()