# System structure



New Class: IncomingItems

import java.io.FileWriter;  
import java.util.Scanner;  
  
public class IncomingItems {  
  
 private String name;  
 private int quantity;  
  
 public IncomingItems() {  
 }  
  
 public IncomingItems(String name, int quantity) {  
 this.name = name;  
 this.quantity = quantity;  
 }  
  
 public String getName() {  
 return name;  
 }  
  
 public void setName(String name) {  
 this.name = name;  
 }  
  
 public int getQuantity() {  
 return quantity;  
 }  
  
 public void setQuantity(int quantity) {  
 this.quantity = quantity;  
 }  
  
 public void Serialize(FileWriter fw){  
 try {  
 fw.write(name);  
 fw.write("\n");  
 fw.write(Integer.toString(quantity));  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
 }  
 public static IncomingItems Deserialize(Scanner fileInput){  
 String n = fileInput.nextLine();  
 int q = Integer.parseInt(fileInput.nextLine());  
 return new IncomingItems(n, q);  
  
 }  
}

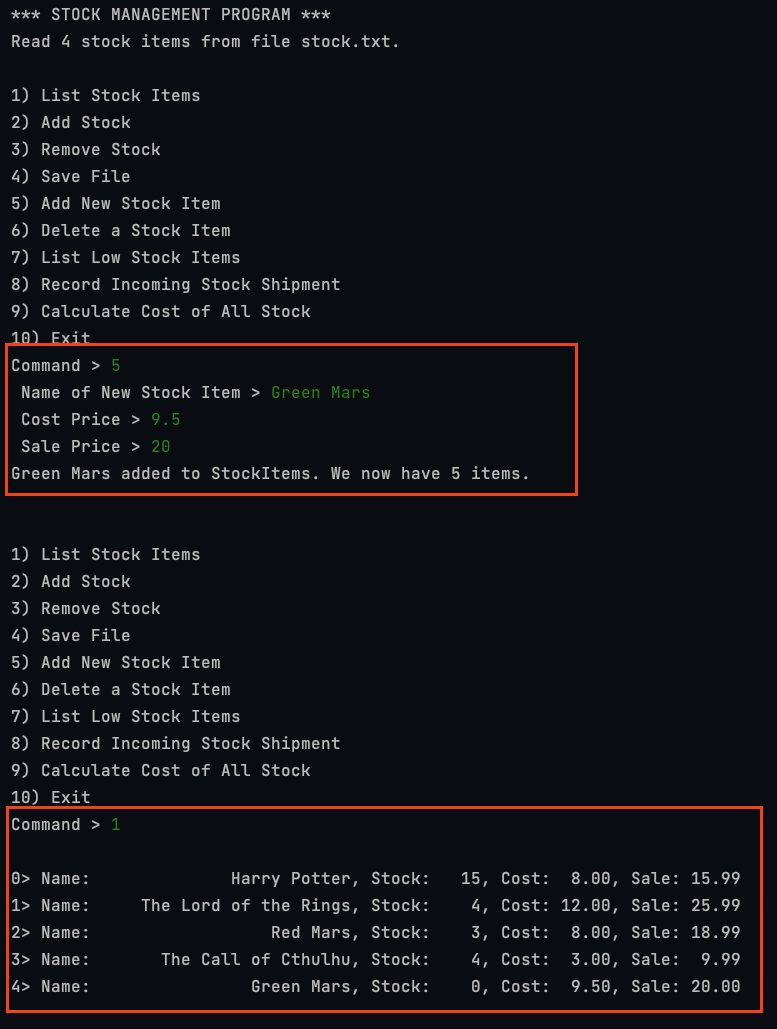
## Function Implement:

5) Add New Stock Item

* Code snippet

private static void AddNewStockItem() {  
 Scanner input = new Scanner(System.in);  
 System.out.printf(" Name of New Stock Item > ");  
 String name = input.nextLine();  
  
 System.out.print(" Cost Price > ");  
 float costPrice = input.nextFloat();  
 System.out.printf(" Sale Price > ");  
 float salePrice = input.nextFloat();  
 StockItem newItem = new StockItem(name, 0, costPrice, salePrice);  
 stockItems.add(newItem);  
 //print out new item information  
 System.out.println( name + " added to StockItems. We now have " + stockItems.size() + " items.");  
  
}

* Output



6) Delete a Stock Item

* Code snippet

private static void DeleteStockItem() {  
 Scanner input = new Scanner(System.in);  
 System.out.printf("\nDelete which Stock Item number? (0-%d) > ", stockItems.size() - 1);  
 int idx = input.nextInt();  
  
 System.out.println("The " + stockItems.get(idx).getName() + " has been deleted.");  
 if (idx >= 0 && idx < stockItems.size()) {  
 stockItems.remove(idx);  
 }  
 System.out.println("We now have " + stockItems.size() + " items.");  
}

* Output



7) List Low Stock Items

* Code snippet

private static void ListLowStockItems() {  
 System.out.printf("We have less than 5 of the following items:");  
 for (int i = 0; i < stockItems.size(); i++) {  
 int stockLevel = stockItems.get(i).getStockLevel();  
 if (stockLevel <= 5) {  
 stockItems.get(i).Print(i);  
 }  
 }  
}

* Output



8) Record Incoming Stock Shipment

* Code snippet

private static boolean RecordIncomingStockShipment() {  
 int recordItemQty = 0;  
 try (Scanner input = new Scanner(Paths.get(incomingItems))) {  
  
 while (input.hasNext()) {  
 //read name and quantity from file: incomingItems  
 String itemName = input.nextLine();  
 int itemQuantity = Integer.parseInt(input.nextLine());  
  
  
 // Find the stock item with the given name  
 boolean found = false;  
 for (int i = 0; i < stockItems.size(); i++) {  
 if (stockItems.get(i).getName().equals(itemName)) {  
 // Update the stock level of the found item  
 stockItems.get(i).ChangeStockLevel(itemQuantity);  
 recordItemQty++;  
 found = true;  
 break;  
 }  
 //print warning if item not found, just print one time  
 if (i == stockItems.size() - 1) {  
 System.out.println("ERROR! No StockItem found with the name:" + itemName);  
 }  
 }  
 }  
 System.out.println("Recorded " + recordItemQty + " items in shipment file");  
 input.close();  
 } catch (IOException e) {  
 System.out.printf("\nCannot load file %s\n", incomingItems);  
 return false;  
 }  
  
 return true;  
  
}

* Output



9) Calculate Cost of All Stock

* Code snippet

private static void CalculateCostofAllStock() {  
 float totalCost = 0;  
 int totalStock = 0;  
 for (int i = 0; i < stockItems.size(); i++) {  
 totalCost += stockItems.get(i).getStockLevel() \* stockItems.get(i).getCostPrice();  
 totalStock += stockItems.get(i).getStockLevel();  
 }  
 //Our total stock is 57 and total cost is 436.00.  
 System.out.printf("Our total stock is %d and total cost is %.2f.", totalStock, totalCost);  
}

* Output

