public static void createOrder() throws IOException {  
 // initialize the needed variables  
 Scanner customerInput = new Scanner(System.*in*);  
 boolean continueToBuy = true;  
 int quantity, itemNum;  
 double chosenItemPrice = 0, total = 0, totalAfterDiscount = 0;  
 HashMap<String, ArrayList<Integer>> orderProducts = new HashMap<>();  
 String chosenProductName = null;  
  
 // loop when customer still want to buy more products  
 while (continueToBuy) {  
 // display all the products  
 System.*out*.println("PRODUCT");  
 System.*out*.println("--------------");  
 *viewProducts*();  
 System.*out*.println("--------------");  
 System.*out*.println("Enter the number of the item you want to buy:");  
 itemNum = customerInput.nextInt();  
  
 File productFile = new File("./src/File/products.txt");  
 Scanner productFileScanner = new Scanner(productFile);  
  
 while (productFileScanner.hasNextLine()) {  
 String product = productFileScanner.nextLine();  
 String productId = product.split(",")[0];  
 int idNumber = Integer.*parseInt*(productId.substring(1, 4));  
 String productName = product.split(",")[1];  
 double productPrice = Double.*parseDouble*(product.split(",")[2]);  
  
 // when the item number that the user enters is equal to the ID number of the product, use that product name and price  
 if (itemNum == idNumber) {  
 chosenItemPrice = productPrice;  
 chosenProductName = productName;  
 break;  
 }  
 }  
  
 System.*out*.println("How many do you want to buy?");  
 quantity = customerInput.nextInt();  
  
 // update hash map with the product and its quantity.  
 // Use array list in case customers want to add the product that is already in their cart  
  
 if (orderProducts.containsKey(chosenProductName)) {  
 ArrayList<Integer> quantityArrayList;  
 quantityArrayList = orderProducts.get(chosenProductName);  
 quantityArrayList.add(quantity);  
 orderProducts.put(chosenProductName, quantityArrayList);  
 } else {  
 ArrayList<Integer> quantityArrayList = new ArrayList<Integer>();  
 quantityArrayList.add(quantity);  
 orderProducts.put(chosenProductName,quantityArrayList);  
 }  
  
 total = total + (chosenItemPrice \* quantity);  
  
 System.*out*.println("Do you want to continue to buy? (Y or N)");  
 String isContinue = customerInput.next();  
  
 // prevent the case customer enters something apart from Y or N.  
 if (isContinue.equalsIgnoreCase("Y"))  
 System.*out*.println("-----------------");  
 else if (isContinue.equalsIgnoreCase("N")) {  
 System.*out*.println("-----------------");  
 continueToBuy = false;  
 } else  
 System.*out*.println("Wrong input! Y or N only! Please try again");  
 }  
  
  
   
 // let the customer enter and confirm the shipping address  
 System.*out*.println("Your shipping address is:");  
 String address = customerInput.next();  
  
 System.*out*.println("Please re-check your address. Is it correct? (Y or N) :");  
 String addressConfirm = customerInput.next();  
 while (true) {  
 if (addressConfirm.equalsIgnoreCase("Y")) {  
 System.*out*.println("-----------------");  
 System.*out*.println("Thank you! You have successfully placed your order");  
 break;  
 }  
  
 if (addressConfirm.equalsIgnoreCase("N")) {  
 System.*out*.println("Please re-enter your address!");  
 System.*out*.println("Your shipping address is:");  
 address = customerInput.next();  
 } else {  
 System.*out*.println("Only enter Y or N");  
 }  
 System.*out*.println("Please re-check your address. Is it correct? (Y or N) :");  
 addressConfirm = customerInput.next();  
 }  
  
 // generate orderID  
 String orderId = newOrderId();  
  
 // order date  
 String orderDate = LocalDate.*now*().toString();  
  
  
 // get the list of all products in the order and their quantities  
 Object[] productsList = orderProducts.keySet().toArray();  
 Object[] quantityList = orderProducts.values().toArray();  
  
 // generate strings to store the list of products and their quantities  
 StringBuilder currentOrderProducts = new StringBuilder(productsList[0].toString());  
 StringBuilder currentOrderProductQuantity = new StringBuilder(quantityList[0].toString());  
 for (int i = 1; i < orderProducts.size(); i++) {  
 currentOrderProducts.append(":").append(productsList[i].toString());  
 }  
 for (int i = 1; i < orderProducts.size(); i++) {  
 currentOrderProductQuantity.append(":").append(quantityList[i].toString());  
 }  
  
 // append the line for the new order to the end of the orders.txt file  
 String newOrder = String.*join*(",", orderId,customerId,username,orderDate,address,  
 currentOrderProducts.toString(), currentOrderProductQuantity.toString(), String.*valueOf*(totalAfterDiscount), "delivered");  
 PrintWriter pw = new PrintWriter(new FileWriter("./src/File/orders.txt",true));  
 pw.println(newOrder);  
  
 pw.flush();  
 pw.close();  
  
 // display order details  
 System.*out*.println("Your order detail");  
 System.*out*.println("-----------------");  
 System.*out*.println("OrderID: " + orderId);  
 System.*out*.println("Order date: "+ orderDate);  
 System.*out*.println("Customer name:"+ username);  
 System.*out*.println("Shipping address: "+ address);  
 System.*out*.println("Items: "+ currentOrderProducts );  
 System.*out*.println("Quantity: "+ currentOrderProductQuantity);  
 System.*out*.println("Total: " + total);  
  
 // update membership by adding the order total to total spending  
  
}

public void createOrder() throws IOException {  
 // initialize the needed variables  
 Scanner customerInput = new Scanner(System.*in*);  
 boolean continueToBuy = true;  
 int quantity;  
 String itemName;  
 double total = 0, totalAfterDiscount = 0;  
 HashMap<String, Integer> orderProducts = new HashMap<>();  
  
 // loop when customer still want to buy more products  
 while (continueToBuy) {  
 // display all the products  
 System.*out*.println("PRODUCT");  
 System.*out*.println("--------------");  
 *viewProducts*();  
 System.*out*.println("--------------");  
 System.*out*.println("Enter the name of the item you want to buy:");  
 itemName = customerInput.nextLine();  
  
 String chosenProductName = null;  
 double chosenItemPrice = 0;  
 boolean productMatched = false;  
  
 while (!productMatched) {  
 File productFile = new File("./src/File/items.txt");  
 Scanner productFileScanner = new Scanner(productFile);  
  
 while (productFileScanner.hasNextLine()) {  
 String product = productFileScanner.nextLine();  
 String productName = product.split(",")[1];  
 double productPrice = Double.*parseDouble*(product.split(",")[2]);  
  
 // when the name that the user entered equals to the name of the product, take that product's name and price  
 if (itemName.equals(productName)) {  
 chosenItemPrice = productPrice;  
 chosenProductName = productName;  
 productMatched = true;  
 }  
 }  
 // notice the user when the input product name was not found  
 if (!productMatched) {  
 System.*out*.println("Can't find the product. Please enter another name.");  
 }  
 }  
  
 quantity = InputValidator.*getIntInput*("How many do you want to buy?\n",  
 "Only enter int number!");  
  
  
 // update hash map with the product and its quantity.  
 // Use if-else condition in case customers want to add the product that is already in their cart  
  
 if (orderProducts.containsKey(chosenProductName)) {  
 orderProducts.put(chosenProductName, quantity + orderProducts.get(chosenProductName));  
 } else {  
 orderProducts.put(chosenProductName, quantity);  
 }  
  
 total = total + (chosenItemPrice \* quantity);  
  
 boolean validInput = false;  
  
 System.*out*.println("Do you want to continue to buy? (Y or N)");  
  
 while (!validInput) {  
 String isContinue = customerInput.nextLine();  
  
 // prevent the case customer enters something apart from Y or N.  
 if (isContinue.equalsIgnoreCase("Y")) {  
 System.*out*.println("-----------------");  
 validInput = true;  
 } else if (isContinue.equalsIgnoreCase("N")) {  
 System.*out*.println("-----------------");  
 continueToBuy = false;  
 validInput = true;  
 } else {  
 System.*out*.println("Wrong input! Y or N only! Please try again");  
  
 System.*out*.println("Do you want to continue to buy? (Y or N)");  
 }  
 }  
 }  
  
 // let the customer enter the shipping address  
 System.*out*.println("Your shipping address is:");  
 String shippingAddress = customerInput.nextLine();  
  
 // generate orderID  
 String orderId = *newOrderId*();  
  
 // order date  
 String orderDate = LocalDate.*now*().toString();  
  
  
 // get the list of all products in the order and their quantities  
 Object[] productsList = orderProducts.keySet().toArray();  
 Object[] quantityList = orderProducts.values().toArray();  
  
 // generate strings to store the list of products and their quantities  
 StringBuilder currentOrderProducts = new StringBuilder(productsList[0].toString());  
 StringBuilder currentOrderProductQuantity = new StringBuilder(quantityList[0].toString());  
 for (int i = 1; i < orderProducts.size(); i++) {  
 currentOrderProducts.append(":").append(productsList[i].toString());  
 }  
 for (int i = 1; i < orderProducts.size(); i++) {  
 currentOrderProductQuantity.append(":").append(quantityList[i].toString());  
 }  
  
 // append the line for the new order to the end of the orders.txt file  
 String newOrder = String.*join*(",", orderId,this.getID(),this.getUsername(),orderDate,shippingAddress,  
 currentOrderProducts.toString(), currentOrderProductQuantity.toString(), String.*valueOf*(totalAfterDiscount), "delivered");  
 PrintWriter pw = new PrintWriter(new FileWriter("./src/File/orders.txt",true));  
 pw.println(newOrder);  
  
 pw.flush();  
 pw.close();  
  
 // display order details  
 System.*out*.println("\nYour order detail");  
 System.*out*.println("-----------------");  
 System.*out*.println("OrderID: " + orderId);  
 System.*out*.println("Order date: "+ orderDate);  
 System.*out*.println("Customer name:"+ this.getUsername());  
 System.*out*.println("Shipping address: "+ shippingAddress);  
 System.*out*.println("Items: "+ currentOrderProducts );  
 System.*out*.println("Quantity: "+ currentOrderProductQuantity);  
 System.*out*.println("Total: " + total);  
  
 // update membership by adding the order total to total spending  
  
}

// apply membership  
if(membership.equals("silver"))  
 totalAfterDiscount = total \* 95/100;  
else if (membership.equals("gold"))  
 totalAfterDiscount = total \* 90/100;  
else if (membership.equals("platinum"))  
 totalAfterDiscount = total \* 85/100;  
else  
 totalAfterDiscount = total;