BASE MENU

Function	#	Description	Sample Input Data	Expected Output	Actual Output	P/F
MainMenu	1	User tries to create a new registration when max users is reached	**nChoice** contains: 1 ***numberofusers** contains: 100	Prints the message "MAX USERS REACHED!"	Prints the message "MAX USERS REACHED!"	P
	2	The user tries to access the user menu	**nChoice** contains: 2	The function userMenu gets called	The function userMenu gets called	P
	3	User tries to access the admin menu	**nChoice** contains: 3	The function AdminMenu gets called	The function AdminMenu gets called	P

Function	#	Description	Sample Input Data	Expected Output	Actual Output	P/F
UserReg	1	All valid inputs	**User ID:** 5 **Password:** Secretpass **Address:** Mondstadt City **ContactNo:** 1101011 **Username:** Chester	**users[*numberofusers].userID contains:** 5 **users[*numberofusers].userPassword contains:** Secretpass **users[*numberofusers].userAddress contains:** Mondstadt City **users[*numberofusers].userContactno contains:** 1101011 **users[*numberofusers].userName contains:** Chester	**users[*numberofusers].userID contains:** 5 **users[*numberofusers].userPassword contains:** Secretpass **users[*numberofusers].userAddress contains:** Mondstadt City **users[*numberofusers].userContactno contains:** 1101011 **users[*numberofusers].userName contains:** Chester	P
	2	User ID has been taken	**User ID:** 5	Prints the message "UserID already taken! Please enter a new one" **users[*numberofusers].userID is not updated**	Prints the message "UserID already taken! Please enter a new one" **users[*numberofusers].userID is not updated**	P
	3	Some strings have spaces	**User ID:** 20 **Password:** Jellyfish **Address:**	**users[*numberofusers].userID contains:** 20	**users[*numberofusers].userID contains:** 20	P

			Watatsumi Island **ContactNo:** 11223 **Username:** Sangonomiya Kokomi	**users[*numberofusers].userPassword contains:** Jellyfish **users[*numberofusers].userAddress contains:** Watatsumi Island **users[*numberofusers].userContactno contains:** 11223 **users[*numberofusers].userName contains:** Sangonomiya Kokomi	**users[*numberofusers].userPassword contains:** Jellyfish **users[*numberofusers].userAddress contains:** Watatsumi Island **users[*numberofusers].userContactno contains:** 11223 **users[*numberofusers].userName contains:** Sangonomiya Kokomi	

Function	**#**	**Description**	**Sample Input Data**	**Expected Output**	**Actual Output**	**P/F**
LoadCart	1	User does not have a cart with their user ID	N/A	Prints the message “No previous carts found”	Prints the message “No previous carts found”	P
	2	User has a cart with their user ID	N/A	cart[] gets updated	cart[] gets updated	P
	3	User has a cart with specific contents	3.txt contains: 5 1 KFC Hashbrown Food staple breakfast food 100 100.00	**cart[0].productID** contains: 5 **cart[0].sellerID** contains: 1 **cart[0].itemName** contains: KFC Hashbrown **cart[0].itemCategory** contains: Food **cart[0].itemDesc** contains: staple breakfast food **cart[0].itemQty** contains: 100 **cart[0].itemPri**	**cart[0].productID** contains: 5 **cart[0].sellerID** contains: 1 **cart[0].itemName** contains: KFC Hashbrown **cart[0].itemCategory** contains: Food **cart[0].itemDesc** contains: staple breakfast food **cart[0].itemQty** contains: 100 **cart[0].itemPri**	P

				ce contains: 100.00	**ce** contains: 100.00	

Function	#	Description	Sample Input Data	Expected Output	Actual Output	P/F
ExitUserMenu	1	User does not have a cart file with their user ID	**users[i].userID** contains: 3	3.txt gets created in the location	3.txt gets created in the location	P
	2	User does not have a cart file with their user ID but has an item in their cart	**cart[0].productID** contains: 5 **cart[0].sellerID** contains: 1 **cart[0].itemName** contains: KFC Hashbrown **cart[0].itemCategory** contains: Food **cart[0].itemDesc** contains: staple breakfast food **cart[0].itemQty** contains: 100 **cart[0].itemPrice** contains: 100.00	3.txt contains: 5 1 KFC Hashbrown Food staple breakfast food 100 100.00	3.txt contains: 5 1 KFC Hashbrown Food staple breakfast food 100 100.00	P
	3	User does not have a cart file with their user ID but has multiple items in their cart	**cart[0].productID** contains: 5 **cart[0].sellerID** contains: 1 **cart[0].itemName** contains: KFC Hashbrown **cart[0].itemCategory** contains: Food **cart[0].itemDesc** contains: staple breakfast food **cart[0].itemQty** contains: 100	3.txt contains: 5 1 KFC Hashbrown [JV] Food staple breakfast food 100 100.00 15 6 Apple [Stan] Food an apple a day.. 22 20.00	3.txt contains: 5 1 KFC Hashbrown [JV] Food staple breakfast food 100 100.00 15 6 Apple [Stan] Food an apple a	P

			cart[0].itemPrice contains: 100.00 **cart[1].productID** contains: 15 **cart[1].sellerID** contains: 6 **cart[1].itemName** contains: Apple **cart[1].itemCategory** contains: Food **cart[1].itemDesc** contains: an apple a day.. **cart[1].itemQty** contains: 22 **cart[1].itemPrice** contains: 20.00		day.. 22 20.00	

Function	**#**	**Description**	**Sample Input Data**	**Expected Output**	**Actual Output**	**P/F**
UserMenu	1	User tries to access the sell menu	**nChoice** contains: 1	SellMenu function gets called	SellMenu function gets called	P
	2	User tries to access the buy menu	**nChoice** contains: 2	LoadCart and BuyMenu gets called	LoadCart and BuyMenu gets called	P
	3	User Exits the menu	**nChoice** contains: 3	ExitUserMenu is called and program returns to the MainMenu	ExitUserMenu is called and program returns to the MainMenu	P

SELL MENU

Function	**#**	**Description**	**Sample Input Data**	**Expected Output**	**Actual Output**	**P/F**
stockMenu	1	User replenishes	**nChoice**	**item->itemQty**	**item->itemQty**	P

		stock of an item	contains: 1 **nNum** contains: 3 **item->itemQty** contains: 1	contains: 4	contains: 4	
	2	User changes the price of an item	**nChoice** contains: 2 **fNum** contains: 125.55 **item->itemPrice** contains: 200.00	**item->itemPrice** contains: 125.55	**item->itemPrice** contains: 125.55	P
	3	User changes the name of an item	**nChoice** contains: 3 **item->itemName** contains: Fish User inputs: Dog	**item->itemName** contains: Dog	**item->itemName** contains: Dog	P
	4	User changes the category of an item	**nChoice** contains: 4 **item->itemCategory** contains: Machine User inputs: 5 star catalyst	**item->itemCategory** contains: 5 star catalyst	**item->itemCategory** contains: 5 star catalyst	P
	5	User changes the description of an item	**nChoice** contains: 5 **item->itemDesc** contains: Best Catalyst User inputs: Cool Catalyst	**item->itemDesc** contains: Cool Catalyst	**item->itemDesc** contains: Cool Catalyst	P

Function	#	Description	Sample Input Data	Expected Output	Actual Output	P/F
editStock	1	User inputs a valid product ID in the users list of items	Logged in user's userID: 1 **nCheckID** contains: 1 **items[0].sellerID** contains: 1 **items[0].productID** contains: 1	Function stockMenu gets called	Function stockMenu gets called	P
	2	User inputs a product ID of a item with a different seller ID	Logged in user's userID: 1 **nCheckID** contains: 7 **items[6].sellerID** contains: 2 **items[6].productID** contains: 7	Prints the message "PRODUCT ID NOT FOUND	Prints the message "PRODUCT ID NOT FOUND	P
	3	User inputs a product ID that is not used by any seller	Logged in user's userID: 1 **nCheckID** contains: 888	Prints the message "PRODUCT ID NOT FOUND	Prints the message "PRODUCT ID NOT FOUND	P

Function	#	Description	Sample Input Data	Expected Output	Actual Output	P/F
AddNewItem	1	Every input is valid	**Input ID:** 40 **Input Name:** Icecream **Input Category:** Food **Input**	**items[0].productID** contains: 40 **items[0].itemName** contains: "Icecream" **items[0].itemCategory** contains:	**items[0].productID** contains: 40 **items[0].itemName** contains: "Icecream" **items[0].itemCategory** contains:	P

			Description: Don't even try **Input Price:** 49 **Input Quantity:** 12	"Food" **items[0].itemDesc** contains: "Don't even try" **items[0].itemQty** contains: 12	"Food" **items[0].itemDesc** contains: "Don't even try" **items[0].itemQty** contains: 12	
	2	Inputted productID is taken	**Input ID:** 40 (productID 40 has been taken prior)	Prints the message "Item Product ID has been taken already!" and asks the user for a new input	Prints the message "Item Product ID has been taken already!" and asks the user for a new input	P
	3	Inputted productID is negative	**InputID:** -11	Prints the message "Item Product ID is Invalid!" and asks the user for a new input	Prints the message "Item Product ID is Invalid!" and asks the user for a new input	P

Function	**#**	**Description**	**Sample Input Data**	**Expected Output**	**Actual Output**	**P/F**
LowStock	1	All products of the user have an itemQty greater than 5	**(These items have the same seller ID)** **items[0].itemQty** contains: 10 **items[1].itemQty** contains: 10 **items[2].itemQty** contains: 20	Prints the message "ALL PRODUCTS ARE PROPERLY STOCKED!"	Prints the message "ALL PRODUCTS ARE PROPERLY STOCKED!"	P
	2	A product of the user has an itemQty of less than 5	**(These items have the same seller ID)** **items[0].itemQty** contains: 10 **items[1].itemQty** contains: 10 **items[2].itemQty** contains: 2	**temp[j]** will have the same contents as item[2], **j** increments.	**temp[j]** will have the same contents as item[2], **j** increments	P
	3	User has no products	**items[]** doesn't contain a single item that has the same sellerID	Prints the message "USER HAS NO PRODUCTS!"	Prints the message "USER HAS NO PRODUCTS!"	P

			as the user's userID			

Function	#	Description	Sample Input Data	Expected Output	Actual Output	P/F
SellMenu	1	User tries to add a new item, but user already has 20 items in the system	**nChoice** contains: 1	Prints the message "Max item limit reached!"	Prints the message "Max item limit reached!"	P
	2	User tries to add a new item and user has less than 20 items in the system	**nChoice** contains: 1	User gets brought to the function AddNewItem	User gets brought to the function AddNewItem	P
	3	User selects the option to edit stock	**nChoice** contains: 2	All products are sorted by productID, then the user gets brought to the editStock menu	All products are sorted by productID, then the user gets brought to the editStock menu	P
	4	User selects the option to show all their products	**nChoice** contains: 3	All products are sorted by productID, then all their products are shown in table format	All products are sorted by productID, then all their products are shown in table format	P
	5	User selects the option to show their low stock products	**nChoice** contains: 4	All of user's low stock products are shown in page format	All of user's low stock products are shown in page format	P
	6	User selects the option to exit sell menu	**nChoice** contains: 5	User gets transported back to UserMenu	User gets transported back to UserMenu	P

BUY MENU

Function	#	**Description**	**Sample Input Data**	**Expected Output**	**Actual Output**	**P/F**
showSpecificSeller	1	User inputs a sellerID that exists	**sellerIDinputfor2** contains: 2	prints all of userID 2's products	prints all of userID 2's products	P
	2	User inputs a sellerID that doesn't exist	**sellerIDinputfor2** contains: 30	Prints "UserID not found!"	Prints "UserID not found!"	P
	3	User inputs an existing sellerID with no products	**sellerIDinputfor2** contains 23	Prints a blank template	Prints a blank template	P

Function	#	**Description**	**Sample Input Data**	**Expected Output**	**Actual Output**	**P/F**
SearchCategory	1	User inputs a category that exists	User inputs the category: 5 star catalyst	**Product ID:** 44 **Item Name:** Moonlight Everglow **Category:** 5 star catalyst **Description:** Really nice ball **Quantity:** 6 **Price:** 6999.99	**Product ID:** 44 **Item Name:** Moonlight Everglow **Category:** 5 star catalyst **Description:** Really nice ball **Quantity:** 6 **Price:** 6999.99	P
	2	User inputs a category that multiple users have the product of	User inputs the category: Art	**Product ID:** 21 **Item Name:** Poem **Category:** Art **Description:** well written poem **Quantity:** 10 **Price:** 1500.00 **Product ID:** 27 **Item Name:** Dance **Category:** Art **Description:** makes u happy **Quantity:** 4 **Price:** 1000.00 **Product ID:** 28 **Item Name:**	**Product ID:** 21 **Item Name:** Poem **Category:** Art **Description:** well written poem **Quantity:** 10 **Price:** 1500.00 **Product ID:** 27 **Item Name:** Dance **Category:** Art **Description:** makes u happy **Quantity:** 4 **Price:** 1000.00 **Product ID:** 28 **Item Name:**	P

				Painting **Category:** Art **Description:** realistic bread **Quantity:** 5 **Price:** 1000.00	Painting **Category:** Art **Description:** realistic bread **Quantity:** 5 **Price:** 1000.00	
	3	User inputs a category that doesn't exist	User inputs the category: Free	Prints the message "NO PRODUCTS TO SHOW!"	Prints the message "NO PRODUCTS TO SHOW!"	P

Function	**#**	**Description**	**Sample Input Data**	**Expected Output**	**Actual Output**	**P/F**
SearchPName	1	User inputs a name that is part of a product's name	User inputs the name: Gravy	**Product ID:** 4 **Item Name:** KFC Gravy Pump **Category:** Machine **Description:** Not stolen **Quantity:** 5 **Price:** 10000.00	**Product ID:** 4 **Item Name:** KFC Gravy Pump **Category:** Machine **Description:** Not stolen **Quantity:** 5 **Price:** 10000.00	P
	2	User inputs a name that multiple products have the name of	User inputs the name: coupon	**Product ID: 8** **Item Name:** FireworkCoupon **Category:** Hopium **Description:** Recipe for firework **Quantity:** 100 **Price:** 10.00 **Product ID:** 10 **Item Name:** PlusCoupon **Category:** Hopium **Description:** Adds points to grade **Quantity:** 20 **Price:** 999.00 **Product ID:** 25 **Item Name:**	**Product ID: 8** **Item Name:** FireworkCoupon **Category:** Hopium **Description:** Recipe for firework **Quantity:** 100 **Price:** 10.00 **Product ID:** 10 **Item Name:** PlusCoupon **Category:** Hopium **Description:** Adds points to grade **Quantity:** 20 **Price:** 999.00 **Product ID:** 25 **Item Name:**	P

				WFP Coupon **Category:** Hopium **Description:** Offers discounts **Quantity:** 100 **Price:** 100.00	WFP Coupon **Category:** Hopium **Description:** Offers discounts **Quantity:** 100 **Price:** 100.00	
	3	User inputs a name that no product has the name of	User inputs the name: beef	Prints the message “NO PRODUCTS TO SHOW!”	Prints the message “NO PRODUCTS TO SHOW!”	P

Function	#	**Description**	**Sample Input Data**	**Expected Output**	**Actual Output**	**P/F**
AddtoCart	1	User inputs a product ID that exists, the product has adequate stock, and is not owned by them	**ProductID** contains: 21 (not owned by buyer) **Quantity** contains: 1 ***itemsincart** contains: 0	**Cart[0].productID** contains: 21 **Cart[0].itemName** contains: Poem **Cart[0].itemCategory** contains: Art **Cart[0].itemDesc** contains: well written poem **Cart[0].itemPrice** contains: 1500.00 **Cart[0].sellerID** contains: 9 **Cart[0].itemQty** contains: 1	**Cart[0].productID** contains: 21 **Cart[0].itemName** contains: Poem **Cart[0].itemCategory** contains: Art **Cart[0].itemDesc** contains: well written poem **Cart[0].itemPrice** contains: 1500.00 **Cart[0].sellerID** contains: 9 **Cart[0].itemQty** contains: 1	P
	2	User inputs a product ID that already exists in their cart	**ProductID** contains: 28 (not owned by buyer, but already exists in the cart)	Prints the message “Item already exists in cart!”	Prints the message “Item already exists in cart!”	P

			Quantity contains: 1 ***itemsincart** contains: 1			
	3	User inputs a product ID that exists, but the product does not have adequate stock	**ProductID** contains: 7 (not owned by buyer) **Quantity** contains: 100 ***itemsincart** contains: 0	Prints the message “Quantity not available!”	Prints the message “Quantity not available!”	P
	4	User inputs a product ID that exists, but that product is owned by them	**ProductID** contains: 33 (owned by buyer) **Quantity** contains: 10 ***itemsincart** contains: 0	Prints the message “Can't buy your own item!”	Prints the message “Can't buy your own item!”	P
	5	User’s cart is full	***itemsincart** contains: 10	Prints the message "Cart is already full! Please consider editing the cart or checking out first before adding more items."	Prints the message "Cart is already full! Please consider editing the cart or checking out first before adding more items."	P
	6	User inputs a product ID that does not exist	**ProductID** contains: 365 (does not exist) **Quantity** contains: 1 ***itemsincart**	Prints the message “Product not found!”	Prints the message “Product not found!”	P

			contains: 0			

Function	#	Description	Sample Input Data	Expected Output	Actual Output	P/F
searchcart	1	Key belongs to an item in the cart with the same sellerID	**key** contains: 1 **cart[0].sellerID** contains: 3 **cart[1].sellerID** contains: 6 **cart[2].sellerID** contains: 1 **Index** contains: -1	**Index** contains: 2	**Index** contains: 2	P
	2	Key belongs to an item not in the cart with the same seller ID	**key** contains: 12 **cart[0].sellerID** contains: 3 **cart[1].sellerID** contains: 6 **cart[2].sellerID** contains: 1 **Index** contains: -1	**Index** contains: -1	**Index** contains: -1	P
	3	Key does not match with any item’s seller ID	**key** contains: 999 **cart[0].sellerID** contains: 3 **cart[1].sellerID** contains: 6 **cart[2].sellerID** contains: 1 **Index**	**Index** contains: -1	**Index** contains: -1	P

			contains: -1			

Function	#	Description	Sample Input Data	Expected Output	Actual Output	P/F
RemoveAllItems	1	User inputs a sellerID that a cart item of theirs own	**Cart[0].productID** contains: 1 **Cart[0].itemName** contains: KFC Spoon **Cart[0].itemCategory** contains: Utensil **Cart[0].itemDesc** contains: it's a spoon **Cart[0].itemPrice** contains: 9999.00 **Cart[0].sellerID** contains: 1 **Cart[0].itemQty** contains: 1 Inputted UserID contains: 1	**Cart[0]** now contains no info	**Cart[0]** now contains no info	P
	2	User inputs a sellerID that multiple cart item of theirs own	**Cart[0].productID** contains: 1 **Cart[0].itemName** contains: KFC Spoon **Cart[0].itemCategory** contains: Utensil **Cart[0].itemDesc** contains: it's a spoon **Cart[0].itemPrice** contains: 9999.00	**Cart[0]** now contains the info previously on Cart[2] (since both Cart[0] and Cart[1]'s sellerIDs were 1)	**Cart[0]** now contains info previously on Cart[2]	P

			Cart[0].sellerID contains: 1 **Cart[0].itemQty** contains: 1 **Cart[1].productID** contains: 2 **Cart[1].itemName** contains: KFC Chicken **Cart[1].itemCategory** contains: Food **Cart[1].itemDesc** contains: kentucky fried **Cart[1].itemPrice** contains: 100.00 **Cart[1].sellerID** contains: 1 **Cart[1].itemQty** contains: 1 **Cart[2].productID** contains: 10 **Cart[2].itemName** contains: PlusCoupon **Cart[2].itemCategory** contains: Hopium **Cart[2].itemDesc** contains: adds points to grade **Cart[2].itemPrice** contains: 999.00 **Cart[2].sellerID** contains: 4 **Cart[2].itemQty** contains: 1			

			Inputted UserID contains: 1			
	3	User inputs a sellerID that none of their cart items own	**Cart[0].productID** contains: 10 **Cart[0].itemName** contains: PlusCoupon **Cart[0].itemCategory** contains: Hopium **Cart[0].itemDesc** contains: adds points to grade **Cart[0].itemPrice** contains: 999.00 **Cart[0].sellerID** contains: 4 **Cart[0].itemQty** contains: 1 Inputted UserID contains: 10	Prints the message “SELLER NOT FOUND!”	Prints the message “SELLER NOT FOUND!”	P
	4	User inputs a sellerID that doesn’t exist	**Cart[0].productID** contains: 10 **Cart[0].itemName** contains: PlusCoupon **Cart[0].itemCategory** contains: Hopium **Cart[0].itemDesc** contains: adds points to grade **Cart[0].itemPrice** contains: 999.00 **Cart[0].sellerID** contains: 4 **Cart[0].itemQty**	Prints the message “SELLER NOT FOUND!”	Prints the message “SELLER NOT FOUND!”	P

			contains: 1 Inputted UserID contains: 28239			

Function	**#**	**Description**	**Sample Input Data**	**Expected Output**	**Actual Output**	**P/F**
RemoveSpecificItem	1	User inputs a productID that a cart item of theirs own (cart only has 1 item)	**Cart[0].productID** contains: 10 **Cart[0].itemName** contains: PlusCoupon **Cart[0].itemCategory** contains: Hopium **Cart[0].itemDesc** contains: adds points to grade **Cart[0].itemPrice** contains: 999.00 **Cart[0].sellerID** contains: 4 **Cart[0].itemQty** contains: 1 Inputted ProductID contains: 10	**Cart[0]** now contains no info	**Cart[0]** now contains no info	P
	2	User inputs a productID that a cart item of theirs own (cart has 3 items)	**Cart[0].productID** contains: 1 **Cart[0].itemName** contains: KFC Spoon **Cart[0].itemCategory** contains: Utensil **Cart[0].itemDesc** contains: it's a spoon	**Cart[0]** now contains the info previously on Cart[1] **Cart[1]** now contains the info previously on Cart[2]	**Cart[0]** now contains the info previously on Cart[1] **Cart[1]** now contains the info previously on Cart[2]	P

			Cart[0].itemPrice contains: 9999.00 **Cart[0].sellerID** contains: 1 **Cart[0].itemQty** contains: 1 **Cart[1].productID** contains: 2 **Cart[1].itemName** contains: KFC Chicken **Cart[1].itemCategory** contains: Food **Cart[1].itemDesc** contains: kentucky fried **Cart[1].itemPrice** contains: 100.00 **Cart[1].sellerID** contains: 1 **Cart[1].itemQty** contains: 1 **Cart[2].productID** contains: 10 **Cart[2].itemName** contains: PlusCoupon **Cart[2].itemCategory** contains: Hopium **Cart[2].itemDesc** contains: adds points to grade **Cart[2].itemPrice** contains: 999.00 **Cart[2].sellerID**			

			contains: 4 **Cart[2].itemQty** contains: 1 Inputted ProductID contains: 1			
	3	User inputs a productID that none of their cart items own	**Cart[0].productID** contains: 1 **Cart[0].itemName** contains: KFC Spoon **Cart[0].itemCategory** contains: Utensil **Cart[0].itemDesc** contains: it's a spoon **Cart[0].itemPrice** contains: 9999.00 **Cart[0].sellerID** contains: 1 **Cart[0].itemQty** contains: 1 Inputted ProductID contains: 2	Prints the message "PRODUCT NOT FOUND!"	Prints the message "SELLER NOT FOUND!"	P
	4	User inputs a productID that doesn't exist	**Cart[0].productID** contains: 1 **Cart[0].itemName** contains: KFC Spoon **Cart[0].itemCategory** contains: Utensil **Cart[0].itemDesc** contains: it's a spoon **Cart[0].itemPrice** contains:	Prints the message "PRODUCT NOT FOUND!"	Prints the message "PRODUCT NOT FOUND!"	P

			9999.00 **Cart[0].sellerID** contains: 1 **Cart[0].itemQty** contains: 1 Inputted ProductID contains: 189			

Function	#	Description	Sample Input Data	Expected Output	Actual Output	P/F
EditQuantity	1	User edits the quantity of a product in their cart into a valid quantity (quantity seller has stock of)	**Cart[0].productID** contains: 1 **Cart[0].itemName** contains: KFC Spoon **Cart[0].itemCategory** contains: Utensil **Cart[0].itemDesc** contains: it's a spoon **Cart[0].itemPrice** contains: 9999.00 **Cart[0].sellerID** contains: 1 **Cart[0].itemQty** contains: 3 Inputted ProductID contains: 1 Inputted Quantity contains: 9	**Cart[0].productID** contains: 1 **Cart[0].itemName** contains: KFC Spoon **Cart[0].itemCategory** contains: Utensil **Cart[0].itemDesc** contains: it's a spoon **Cart[0].itemPrice** contains: 9999.00 **Cart[0].sellerID** contains: 1 **Cart[0].itemQty** contains: 9	**Cart[0].productID** contains: 1 **Cart[0].itemName** contains: KFC Spoon **Cart[0].itemCategory** contains: Utensil **Cart[0].itemDesc** contains: it's a spoon **Cart[0].itemPrice** contains: 9999.00 **Cart[0].sellerID** contains: 1 **Cart[0].itemQty** contains: 9	P
	2	User edits the quantity of a product in their cart into a	**Cart[0].productID** contains: 1 **Cart[0].itemNa**	Prints the message "NOT ENOUGH	Prints the message "NOT ENOUGH	P

		quantity that exceeds the seller's stock	**me** contains: KFC Spoon **Cart[0].itemCategory** contains: Utensil **Cart[0].itemDesc** contains: it's a spoon **Cart[0].itemPrice** contains: 9999.00 **Cart[0].sellerID** contains: 1 **Cart[0].itemQty** contains: 9 Inputted ProductID contains: 1 Inputted Quantity contains: 100	STOCK!"	STOCK!"	
	3	User edits the quantity of a product that does not exist/a product that is not in their cart	Inputted ProductID contains: 10000 Inputted Quantity contains: 1 (productID does not exist)	Prints the message "PRODUCT NOT FOUND!"	Prints the message "PRODUCT NOT FOUND!"	P

Function	**#**	**Description**	**Sample Input Data**	**Expected Output**	**Actual Output**	**P/F**
EditCart	1	User selects the option to remove all items from seller	**nChoice** contains: 1	User gets brought to the function RemoveAllItems	User gets brought to the function RemoveAllItems	P
	2	User selects the option to a	**nChoice** contains: 2	User gets brought to the	User gets brought to	P

		remove specific item		function RemoveSpecificItem	the function RemoveSpecificItem	
	3	User selects the option to edit quantity	**nChoice** contains: 3	User gets brought to the function EditQuantity	User gets brought to the function EditQuantity	P

Function	**#**	**Description**	**Sample Input Data**	**Expected Output**	**Actual Output**	**P/F**
CheckOutAll	1	Cart has the item of one seller	**person.userID** contains: 20 **purchasedate.month** contains: 1 **purchasedate.day** contains: 1 **purchasedate.year** contains: 2022 ***itemsincart** contains: 1 **Cart[0].productID** contains: 1 **Cart[0].itemName** contains: KFC Spoon **Cart[0].itemCategory** contains: Utensil **Cart[0].itemDesc** contains: it's a spoon **Cart[0].itemPrice** contains: 9999.00 **Cart[0].sellerID** contains: 1	Prints the buyerID, date, sellerID, seller name, total transaction price, as well as information on the purchase in table format. **receipt[0].items[0]** contains info on the product in cart[0] ***receiptAmount** contains: 1 ***itemsincart** contains: 0 The stock of productID 1 decreases by the quantity in the cart (5 in this case) Transactions.txt also gets updated to include the information on the purchase	Prints the buyerID, date, sellerID, seller name, total transaction price, as well as information on the purchase in table format. **receipt[0].items[0]** contains info on the product in cart[0] ***receiptAmount** contains: 1 ***itemsincart** contains: 0 The stock of productID 1 decreases by the quantity in the cart (5 in this case) Transactions.txt also gets updated to include the information on the purchase	P

			Cart[0].itemQty contains: 5 ***receiptAmount** contains: 0			
	2	Cart has 2 items from one seller, 1 item from another seller	**person.userID** contains: 20 **purchasedate.month** contains: 1 **purchasedate.day** contains: 1 **purchasedate.year** contains: 2022 ***itemsincart** contains: 3 **Cart[0].productID** contains: 1 **Cart[0].itemName** contains: KFC Spoon **Cart[0].itemCategory** contains: Utensil **Cart[0].itemDesc** contains: it's a spoon **Cart[0].itemPrice** contains: 9999.00 **Cart[0].sellerID** contains: 1 **Cart[0].itemQty** contains: 5 **Cart[1].productID** contains: 2 **Cart[1].itemName** contains: KFC Chicken	Prints the buyerID, date, sellerID, seller name, total transaction price, as well as information on the purchase in table format. (separate tables/receipts per seller) **receipt[0].items[0]** contains info on the product in cart[0] **receipt[0].items[1]** contains info on the product in cart[1] **receipt[1].items[0]** contains info on the product in cart[2] ***receiptAmount** contains: 2 ***itemsincart** contains: 0 The stock of productID 1 decreases by the 5, stock of productID 2 decreases by 3, stock of productID 7 decreases by 5 Transactions.tx	Prints the buyerID, date, sellerID, seller name, total transaction price, as well as information on the purchase in table format. (separate tables/receipts per seller) **receipt[0].items[0]** contains info on the product in cart[0] **receipt[0].items[1]** contains info on the product in cart[1] **receipt[1].items[0]** contains info on the product in cart[2] ***receiptAmount** contains: 2 ***itemsincart** contains: 0 The stock of productID 1 decreases by the 5, stock of productID 2 decreases by 3, stock of productID 7 decreases by 5 Transactions.tx	P

			Cart[1].itemCategory contains: Food **Cart[1].itemDesc** contains: kentucky fried **Cart[1].itemPrice** contains: 100.00 **Cart[1].sellerID** contains: 1 **Cart[1].itemQty** contains: 3 **Cart[2].productID** contains: 7 **Cart[2].itemName** contains: Fireworks **Cart[2].itemCategory** contains: Explosives **Cart[2].itemDesc** contains: Fleeting joy **Cart[2].itemPrice** contains: 100.00 **Cart[2].sellerID** contains: 2 **Cart[2].itemQty** contains: 5 ***receiptAmount** contains: 0	t also gets updated to include the information on the purchases, with each different seller having different receipts	t also gets updated to include the information on the purchases, with each different seller having different receipts	
	3	Cart has no items	**person.userID** contains: 20 **purchasedate.month** contains: 1 **purchasedate.day** contains: 1	Nothing happens	Nothing happens	P

			purchasedate.year contains: 2022 ***itemsincart** contains: 3 ***receiptAmount** contains: 0			

Function	#	Description	Sample Input Data	Expected Output	Actual Output	P/F
CheckOutSeller	1	One item in cart has the sellerID	**tempsellerID** contains: 1 **person.userID** contains: 20 **purchasedate.month** contains: 1 **purchasedate.day** contains: 1 **purchasedate.year** contains: 2022 ***itemsincart** contains: 3 **Cart[0].productID** contains: 1 **Cart[0].itemName** contains: KFC Spoon **Cart[0].itemCategory** contains: Utensil **Cart[0].itemDesc** contains: it's a spoon **Cart[0].itemPrice** contains: 9999.00	Prints the buyerID, date, sellerID, seller name, total transaction price, as well as information on the purchase in table format. **receipt[0].items[0]** contains info on the product in cart[0] ***receiptAmount** contains: 1 ***itemsincart** contains: 0 The stock of productID 1 decreases by the quantity in the cart (5 in this case) Transactions.txt also gets updated to include the information on the purchase	Prints the buyerID, date, sellerID, seller name, total transaction price, as well as information on the purchase in table format. **receipt[0].items[0]** contains info on the product in cart[0] ***receiptAmount** contains: 1 ***itemsincart** contains: 0 The stock of productID 1 decreases by the quantity in the cart (5 in this case) Transactions.txt also gets updated to include the information on the purchase	P

			Cart[0].sellerID contains: 1 **Cart[0].itemQty** contains: 5			
	2	Multiple items in cart has the sellerID	**tempsellerID** contains: 1 **person.userID** contains: 20 **purchasedate.month** contains: 1 **purchasedate.day** contains: 1 **purchasedate.year** contains: 2022 ***itemsincart** contains: 3 **Cart[0].productID** contains: 1 **Cart[0].itemName** contains: KFC Spoon **Cart[0].itemCategory** contains: Utensil **Cart[0].itemDesc** contains: it's a spoon **Cart[0].itemPrice** contains: 9999.00 **Cart[0].sellerID** contains: 1 **Cart[0].itemQty** contains: 5 **Cart[1].productID** contains: 7	Prints the buyerID, date, sellerID, seller name, total transaction price, as well as information on the purchase in table format. **receipt[0].items[0]** contains info on the product in cart[0] **receipt[0].items[1]** contains info on the product in cart[2] ***receiptAmount** contains: 1 ***itemsincart** contains: 1 The stock of productID 1 decreases by the 5, stock of productID 2 decreases by 3 Transactions.txt also gets updated to include the information on the purchase	Prints the buyerID, date, sellerID, seller name, total transaction price, as well as information on the purchase in table format. **receipt[0].items[0]** contains info on the product in cart[0] **receipt[0].items[1]** contains info on the product in cart[2] ***receiptAmount** contains: 1 ***itemsincart** contains: 1 The stock of productID 1 decreases by the 5, stock of productID 2 decreases by 3 Transactions.txt also gets updated to include the information on the purchase	P

			Cart[1].itemName contains: Fireworks **Cart[1].itemCategory** contains: Explosives **Cart[1].itemDesc** contains: Fleeting joy **Cart[1].itemPrice** contains: 100.00 **Cart[1].sellerID** contains: 2 **Cart[1].itemQty** contains: 5 **Cart[2].productID** contains: 2 **Cart[2].itemName** contains: KFC Chicken **Cart[2].itemCategory** contains: Food **Cart[2].itemDesc** contains: kentucky fried **Cart[2].itemPrice** contains: 100.00 **Cart[2].sellerID** contains: 1 **Cart[2].itemQty** contains: 3 ***receiptAmount** contains: 0			
	3	No item in cart has the sellerID	**tempsellerID** contains: 1 **person.userID** contains: 20	Nothing happens	Nothing happens	P

			purchasedate.month contains: 1 **purchasedate.day** contains: 1 **purchasedate.year** contains: 2022 ***itemsincart** contains: 1 **Cart[0].productID** contains: 7 **Cart[0].itemName** contains: Fireworks **Cart[0].itemCategory** contains: Explosives **Cart[0].itemDesc** contains: Fleeting joy **Cart[0].itemPrice** contains: 100.00 **Cart[0].sellerID** contains: 2 **Cart[0].itemQty** contains: 5 ***receiptAmount** contains: 0			
	4	There are no items in cart	**tempsellerID** contains: 1 **person.userID** contains: 20 **purchasedate.month** contains: 1 **purchasedate.day** contains: 1	Nothing happens	Nothing happens	P

			purchasedate.year contains: 2022 ***itemsincart** contains: 0 ***receiptAmount** contains: 0			

Function	#	Description	Sample Input Data	Expected Output	Actual Output	P/F
CheckOutItem	1	User enters a product ID that does not exist	**tempprodID** contains: -5	Prints the message “PRODUCT NOT FOUND!”	Prints the message “PRODUCT NOT FOUND!”	P
	2	User enters a product ID within the cart	**tempprodID** contains: 23 **person.userID** contains: 20 **purchasedate** contains: 1 1 2018 **cart[1].productID** contains: 23 **cart[1].sellerID** contains: 11 **cart[1].itemPrice** contains: 10.00 **cart[1].itemQty** contains: 3 (sellerID of the item matches userID) otherpeople[10].userName contains: Micole **items[22].itemQty** contains:	**receipt[0].dates** contains: 1 1 2018 **receipt[0].buyerID** contains: 20 **receipt[0].sellerID** contains: 11 **receipt[0].transactAmount** contains: 30.00 **receipt[0].items[0]** contains: cart[1] **sellerName** contains: Micole **items[22].itemQty** contains: 195 Transactions.txt also gets updated to include the	**receipt[0].dates** contains: 1 1 2018 **receipt[0].buyerID** contains: 20 **receipt[0].sellerID** contains: 11 **receipt[0].transactAmount** contains: 30.00 **receipt[0].items[0]** contains: cart[1] **sellerName** contains: Micole **items[22].itemQty** contains: 195 Transactions.txt also gets updated to include the	P

			200 **receipt[0].items [0].itemQty** contains: 5	information on the purchase	information on the purchase	
	3	User inputs a product ID that exists, but not within the cart	**tempprodID** contains: 5 **person.userID** contains: 20 **purchasedate** contains: 1 1 2018 **cart[1].productID** contains: 23 **cart[1].sellerID** contains: 11 **cart[1].itemPrice** contains: 10.00 **cart[1].itemQty** contains: 3 (sellerID of the item matches userID) otherpeople[10]. userName contains: Micole **items[22].item Qty** contains: 200 **receipt[0].items [0].itemQty** contains: 5	Prints the message “PRODUCT NOT FOUND!”	Prints the message “PRODUCT NOT FOUND!”	P

Function	#	Description	Sample Input Data	Expected Output	Actual Output	P/F
CheckOut Menu	1	Loaded cart’s price is the same as items[]’s price	**cart[0].productID** contains: 1 **cart[0].itemQty** contains: 5	If **nChoice** contains: 1, user gets brought to	If **nChoice** contains: 1, user gets brought to	P

		and cart's quantity doesn't exceed current available stock	**cart[0].itemPrice** contains: 9999.00 **items[0].productID** contains: 1 **items[0].itemQty** contains: 10 **items[0].itemPrice** contains: 9999.00 **purchasedate.month** contains: 1 **purchasedate.day** contains: 1 **purchasedate.year** contains: 2022 **nChoice** contains a value inputted by the user	CheckOutAll If **nChoice** contains: 2, user gets brought to CheckOutSeller If **nChoice** contains: 3, user gets brought to CheckOutItem If **nChoice** contains: 4, user exits CheckOutmenu and gets brought back to BuyMenu If **nChoice** contains a value outside of 1-4, user gets asked to input valid value	CheckOutAll If **nChoice** contains: 2, user gets brought to CheckOutSeller If **nChoice** contains: 3, user gets brought to CheckOutItem If **nChoice** contains: 4, user exits CheckOutmenu and gets brought back to BuyMenu If **nChoice** contains a value outside of 1-4, user gets asked to input valid value	
	2	Loaded cart's price is the same as items[]'s price but cart's quantity exceeds current available stock	**cart[0].productID** contains: 1 **cart[0].itemQty** contains: 5 **cart[0].itemPrice** contains: 9999.00 **items[0].productID** contains: 1 **items[0].itemQty** contains: 4 **items[0].itemPrice** contains: 9999.00 **purchasedate.month** contains: 1 **purchasedate.day** contains: 1 **purchasedate.year** contains: 2022	Prints the message "SELLER HAS UPDATED ITEM 'KFC Spoon's PRICE AND/OR QUANTITY NEW PRICE: 9999.00 NEW QUANTITY: 4 You can still edit your cart if you want to make changes!" **cart[0].itemQty** contains: 4	Prints the message "SELLER HAS UPDATED ITEM 'KFC Spoon's PRICE AND/OR QUANTITY NEW PRICE: 9999.00 NEW QUANTITY: 4 You can still edit your cart if you want to make changes!" **cart[0].itemQty** contains: 4	P

			nChoice contains a value inputted by the user	**cart[0].itemPrice** contains: 9999.00 If **nChoice** contains: 1, user gets brought to CheckOutAll If **nChoice** contains: 2, user gets brought to CheckOutSeller If **nChoice** contains: 3, user gets brought to CheckOutItem If **nChoice** contains: 4, user exits CheckOutmenu and gets brought back to BuyMenu If **nChoice** contains a value outside of 1-4, user gets asked to input valid value	**cart[0].itemPrice** contains: 9999.00 If **nChoice** contains: 1, user gets brought to CheckOutAll If **nChoice** contains: 2, user gets brought to CheckOutSeller If **nChoice** contains: 3, user gets brought to CheckOutItem If **nChoice** contains: 4, user exits CheckOutmenu and gets brought back to BuyMenu If **nChoice** contains a value outside of 1-4, user gets asked to input valid value	
	3	Loaded cart's price is not the same as items[]'s price and cart's quantity doesn't exceed current available stock	**cart[0].productID** contains: 1 **cart[0].itemQty** contains: 5 **cart[0].itemPrice** contains: 9999.00 **items[0].productID** contains: 1 **items[0].itemQty** contains: 10 **items[0].itemPrice** contains:	Prints the message "SELLER HAS UPDATED ITEM 'KFC Spoon's PRICE AND/OR QUANTITY NEW PRICE: 8888.00 NEW QUANTITY: 10	Prints the message "SELLER HAS UPDATED ITEM 'KFC Spoon's PRICE AND/OR QUANTITY NEW PRICE: 8888.00 NEW QUANTITY: 10	P

			8888.00 **purchasedate.month** contains: 1 **purchasedate.day** contains: 1 **purchasedate.year** contains: 2022 **nChoice** contains a value inputted by the user	You can still edit your cart if you want to make changes!” **cart[0].itemPrice** contains: 8888.00 If **nChoice** contains: 1, user gets brought to CheckOutAll If **nChoice** contains: 2, user gets brought to CheckOutSeller If **nChoice** contains: 3, user gets brought to CheckOutItem If **nChoice** contains: 4, user exits CheckOutmenu and gets brought back to BuyMenu If **nChoice** contains a value outside of 1-4, user gets asked to input valid value	You can still edit your cart if you want to make changes!” **cart[0].itemPrice** contains: 8888.00 If **nChoice** contains: 1, user gets brought to CheckOutAll If **nChoice** contains: 2, user gets brought to CheckOutSeller If **nChoice** contains: 3, user gets brought to CheckOutItem If **nChoice** contains: 4, user exits CheckOutmenu and gets brought back to BuyMenu If **nChoice** contains a value outside of 1-4, user gets asked to input valid value	
	4	Loaded cart’s price is not the same as items[]’s price and cart’s quantity exceeds current available stock	**cart[0].productID** contains: 1 **cart[0].itemQty** contains: 5 **cart[0].itemPrice** contains: 9999.00 **items[0].productI**	Prints the message “SELLER HAS UPDATED ITEM 'KFC Spoon's PRICE AND/OR QUANTITY	Prints the message “SELLER HAS UPDATED ITEM 'KFC Spoon's PRICE AND/OR QUANTITY	P

			D contains: 1 **items[0].itemQty** contains: 4 **items[0].itemPrice** contains: 8888.00 **purchasedate.month** contains: 1 **purchasedate.day** contains: 1 **purchasedate.year** contains: 2022 **nChoice** contains a value inputted by the user	NEW PRICE: 8888.00 NEW QUANTITY: 4 You can still edit your cart if you want to make changes!" **cart[0].itemQty** contains: 4 **cart[0].itemPrice** contains: 8888.00 If **nChoice** contains: 1, user gets brought to CheckOutAll If **nChoice** contains: 2, user gets brought to CheckOutSeller If **nChoice** contains: 3, user gets brought to CheckOutItem If **nChoice** contains: 4, user exits CheckOutmenu and gets brought back to BuyMenu If **nChoice** contains a value outside of 1-4, user gets asked to input valid value	NEW PRICE: 8888.00 NEW QUANTITY: 4 You can still edit your cart if you want to make changes!" **cart[0].itemQty** contains: 4 **cart[0].itemPrice** contains: 8888.00 If **nChoice** contains: 1, user gets brought to CheckOutAll If **nChoice** contains: 2, user gets brought to CheckOutSeller If **nChoice** contains: 3, user gets brought to CheckOutItem If **nChoice** contains: 4, user exits CheckOutmenu and gets brought back to BuyMenu If **nChoice** contains a value outside of 1-4, user gets asked to input valid value	

Function	#	Description	Sample Input Data	Expected Output	Actual Output	P/F
BuyMenu	1	User selects the option to view all products	**nChoice** contains: 1	UserIDs are sorted, products of each user is shown page by page	UserIDs are sorted, products of each user is shown page by page	P
	2	User selects the option to view a specific seller's products	**nChoice** contains: 2	Program asks the user for a sellerID input, then calls ShowSpecific Seller	Program asks the user for a sellerID input, then calls ShowSpecific Seller	P
	3	User selects the option to search product by category	**nChoice** contains: 3	User gets transported to SearchCateg ory	User gets transported to SearchCateg ory	P
	4	User selects the option to search product by name	**nChoice** contains: 4	User gets transported to SearchPNam e	User gets transported to SearchPNam e	P
	5	User selects the option to add to cart	**nChoice** contains: 5	User gets transported to AddtoCart	User gets transported to AddtoCart	P
	6	User selects the option to edit cart	**nChoice** contains: 6	User gets transported to EditCart	User gets transported to EditCart	P
	7	User selects the option to go to the checkout menu	**nChoice** contains: 7	User gets transported to CheckOutMe nu	User gets transported to CheckOutMe nu	P
	8	User selects the option to exit	**nChoice** contains: 8	Program transports user back to main menu	Program transports user back to main menu	P
	9	User inputs an invalid number	**nChoice** contains: 9	Program asks user for input again	Program asks user for input again	P

ADMIN MENU

Function	#	**Description**	**Sample Input Data**	**Expected Output**	**Actual Output**	**P/F**
checkDate Range	1	Date Input is has greater than start date year and less than end date year	**Transactions[i].dates** contains the date 4/7/2023 **startDate** contains 1/1/2020 **endDate** contains: 2/2/2024 **WithinDate** contains: 0	**WithinDate** contains: 1	**WithinDate** contains: 1	P
	2	Date Input is has equal years and within the month range	**Transactions[i].dates** contains the date 4/7/2023 **startDate** contains 3/1/2023 **endDate** contains: 5/2/2023 **WithinDate** contains: 0	**WithinDate** contains: 1	**WithinDate** contains: 1	P
	3	Date Input is has equal years and equal months and within the day range	**Transactions[i].dates** contains the date 4/7/2023 **startDate** contains 4/1/2023 **endDate** contains: 4/8/2023 **WithinDate** contains: 0	**WithinDate** contains: 1	**WithinDate** contains: 1	P
	4	Date Input is has equal year as	**Transactions[i].dates** contains	**WithinDate** contains: 1	**WithinDate** contains: 1	P

		startDate, month is greater than startDate and within the range	the date 4/7/2023 **startDate** contains 3/1/2023 **endDate** contains: 4/8/2024 **WithinDate** contains: 0			
	5	Date Input is has equal year as endDate, month is less than endDate and within the range	**Transactions[i].dates** contains the date 4/7/2023 **startDate** contains 9/1/2022 **endDate** contains: 5/1/2023 **WithinDate** contains: 0	**WithinDate** contains: 1	**WithinDate** contains: 1	P
	6	Date Input is greater than endDate year	**Transactions[i].dates** contains the date 4/7/2023 **startDate** contains 9/1/2021 **endDate** contains: 5/1/2022 **WithinDate** contains: 0	**WithinDate** contains: 0	**WithinDate** contains: 0	P

Function	**#**	**Description**	**Sample Input Data**	**Expected Output**	**Actual Output**	**P/F**
getTransaction	1	Transactions.txt does not exist yet	N/A	Prints the message “Cannot open	Prints the message “Cannot open	P

				file.”	file.”	
	2	Transactions.txt exists but does not contain data		Nothing changes	Nothing changes	P
	3	Transactions.txt exists and contains data	***transNo** contains: 0 Transactions.txt contains: 20 4 7 2023 Fireworks [Yoi] 22 7 100.00 = 2 Naganohara Yoimiya 2200.00 -	**transactions[0].buyerID** contains: 20 **transactions[0].dates.month** contains: 4 **transactions[0].dates.day** contains: 7 **transactions[0].dates.year** contains: 2023 **transactions[0].items[0].itemName** contains: Fireworks [Yoi] **transactions[0].items[0].itemQty** contains: 22 **transactions[0].items[0].productID** contains: 7 **transactions[0].items[0].itemPrice** contains: 100.00 **transactions[0].sellerID** contains: 2 **String** contains: Naganohara Yoimiya **transactions[0].transactAmount** contains: 2200.00	**transactions[0].buyerID** contains: 20 **transactions[0].dates.month** contains: 4 **transactions[0].dates.day** contains: 7 **transactions[0].dates.year** contains: 2023 **transactions[0].items[0].itemName** contains: Fireworks [Yoi] **transactions[0].items[0].itemQty** contains: 22 **transactions[0].items[0].productID** contains: 7 **transactions[0].items[0].itemPrice** contains: 100.00 **transactions[0].sellerID** contains: 2 **String** contains: Naganohara Yoimiya **transactions[0].transactAmount** contains: 2200.00	P

Function	**#**	**Description**	**Sample Input Data**	**Expected Output**	**Actual Output**	**P/F**

ShowTotal Sales	1	Start and end date encapsulates all the transactions	**Start date** input: 5/2/2021 **End date** input: 2/24/2024 **Date**: 4/7/2023 **transactions[0].transactAmount** contains: 2200.00 **Date**: 2/22/2024 **transactions[1].transactAmount** contains: 10588.00 **transactions[2].transactAmount** contains: 440.00 **Date**: 5/12/2021 **transactions[3].transactAmount** contains: 25000.00 **transactions[4].transactAmount** contains: 20.00 **totalTransaction** contains: 0.00	**totalTransaction** contains: 38248.00	**totalTransaction** contains: 38248.00	P
	2	Start and end date encapsulates two of the transactions	**Start date** input: 5/2/2021 **End date** input: 7/22/2022 **Date**: 4/7/2023 **transactions[0].transactAmount** contains: 2200.00 **Date**: 2/22/2024 **transactions[1].transactAmount** contains: 10588.00 **transactions[2].transactAmount** contains: 440.00 **Date**: 5/12/2021 **transactions[3].transactAmount** contains: 25000.00 **transactions[4].transactAmount** contains: 20.00 **totalTransaction**	**totalTransaction** contains: 25020.00	**totalTransaction** contains: 25020.00	P

			contains: 0.00			
	3	Start and end date encapsulates none of the transactions	**Start date** input: 5/2/2025 **End date** input: 7/22/2026 **Date**: 4/7/2023 **transactions[0].transactAmount** contains: 2200.00 **Date**: 2/22/2024 **transactions[1].transactAmount** contains: 10588.00 **transactions[2].transactAmount** contains: 440.00 **Date**: 5/12/2021 **transactions[3].transactAmount** contains: 25000.00 **transactions[4].transactAmount** contains: 20.00 **totalTransaction** contains: 0.00	**totalTransaction** contains: 0.00	**totalTransaction** contains: 0.00	P

Function	**#**	**Description**	**Sample Input Data**	**Expected Output**	**Actual Output**	**P/F**
ShowSellersSales	1	Date input encapsulates all transactions	**Start date** input: 5/2/2021 **End date** input: 2/24/2024 **Date**: 4/7/2023 **transactions[0].sellerID** contains: 2(index 1 in users array) **transactions[0].transactAmount** contains: 2200.00 **Date**: 2/22/2024 **transactions[1].sellerID** contains: 1 (index 0) **transactions[1].transactAmount** contains: 10588.00	**totalTransaction[0]** contains: 10588.00 **totalTransaction[1]** contains: 2200.00 **totalTransaction[5]** contains: 440.00 **totalTransaction[8]** contains: 25000.00 **totalTransaction[10]** contains: 20.00	**totalTransaction[0]** contains: 10588.00 **totalTransaction[1]** contains: 2200.00 **totalTransaction[5]** contains: 440.00 **totalTransaction[8]** contains: 25000.00 **totalTransaction[10]** contains: 20.00	P

			transactions[2].sellerID contains: 6 (index 5) **transactions[2].transactAmount** contains: 440.00 **Date**: 5/12/2021 **transactions[3].sellerID** contains: 9 (index 8) **transactions[3].transactAmount** contains: 25000.00 **transactions[4].sellerID** contains: 11 (index 10) **transactions[4].transactAmount** contains: 20.00 **totalTransaction[0]** contains: 0.00 **totalTransaction[1]** contains: 0.00 **totalTransaction[5]** contains: 0.00 **totalTransaction[8]** contains: 0.00 **totalTransaction[10]** contains: 0.00			
	2	Date input encapsulates multiple transactions with the same seller ID	**Start date** input: 9/24/2023 **End date** input: 2/24/2027 **Date**: 4/7/2023 **transactions[0].sellerID** contains: 2(index 1 in users array) **transactions[0].transactAmount** contains: 2200.00 **Date**: 2/22/2024 **transactions[1].sellerID** contains: 1 (index 0) **transactions[1].transactAmount** contains: 10588.00 **transactions[2].sellerID** contains: 6 (index 5) **transactions[2].transactAmount** contains: 440.00 **Date**: 5/12/2021 **transactions[3].sellerID** contains: 9 (index 8)	**totalTransaction[0]** contains: 10832.00 **totalTransaction[2]** contains: 100000.00 **totalTransaction[5]** contains: 440.00	**totalTransaction[0]** contains: 10832.00 **totalTransaction[2]** contains: 100000.00 **totalTransaction[5]** contains: 440.00	

			transactions[3].transactAmount contains: 25000.00 **transactions[4].sellerID** contains: 11 (index 10) **transactions[4].transactAmount** contains: 20.00 **Date**: 9/25/2023 **transactions[5].sellerID** contains: 1 (index 0) **transactions[5].transactAmount** contains: 244.00 **transactions[6].sellerID** contains: 3 (index 2) **transactions[6].transactAmount** contains: 100000.00 **totalTransaction[0]** contains: 0.00 **totalTransaction[2]** contains: 0.00 **totalTransaction[5]** contains: 0.00			
	3	Date does not encapsulate any transaction	**Start date** input: 9/24/2026 **End date** input: 2/24/2027 **Date**: 4/7/2023 **transactions[0].sellerID** contains: 2(index 1 in users array) **transactions[0].transactAmount** contains: 2200.00 **Date**: 2/22/2024 **transactions[1].sellerID** contains: 1 (index 0) **transactions[1].transactAmount** contains: 10588.00 **transactions[2].sellerID** contains: 6 (index 5) **transactions[2].transactAmount** contains: 440.00 **Date**: 5/12/2021 **transactions[3].sellerID** contains: 9 (index 8) **transactions[3].transact**	Each element in **totalTransaction** array contains: 0.00	Each element in **totalTransaction** array contains: 0.00	P

			Amount contains: 25000.00 **transactions[4].sellerID** contains: 11 (index 10) **transactions[4].transactAmount** contains: 20.00 **Date**: 9/25/2023 **transactions[5].sellerID** contains: 1 (index 0) **transactions[5].transactAmount** contains: 244.00 **transactions[6].sellerID** contains: 3 (index 2) **transactions[6].transactAmount** contains: 100000.00 **totalTransaction[0]** contains: 0.00			

Function	#	Description	Sample Input Data	Expected Output	Actual Output	P/F
ShowShopaholics	1	Date inputs encapsulate all transactions	**Start date** input: 5/2/2021 **End date** input: 2/24/2024 **Date**: 4/7/2023 **transactions[0].buyerID** contains: 20 (index 19 in users array) **transactions[0].transactAmount** contains: 2200.00 **Date**: 2/22/2024 **transactions[1].buyerID** contains: 3 (index 2) **transactions[1].transactAmount** contains: 10588.00 **transactions[2].buyerID** contains: 3 (index 2) **transactions[2].transactAmount**	**totalTransaction[0]** contains: 25020.00 **totalTransaction[2]** contains: 11028.00 **totalTransaction[3]** contains: 100244.00 **totalTransaction[19]** contains: 2200.00	**totalTransaction[0]** contains: 25020.00 **totalTransaction[2]** contains: 11028.00 **totalTransaction[3]** contains: 100244.00 **totalTransaction[19]** contains: 2200.00	P

			contains: 440.00 **Date**: 5/12/2021 **transactions[3].buyerID** contains: 1 (index 0) **transactions[3].transactAmount** contains: 25000.00 **transactions[4].buyerID** contains: 1 (index 0) **transactions[4].transactAmount** contains: 20.00 **Date**: 9/25/2023 **transactions[5].buyerID** contains: 4 (index 3) **transactions[5].transactAmount** contains: 244.00 **transactions[6].buyerID** contains: 4 (index 3) **transactions[6].transactAmount** contains: 100000.00 **totalTransaction[0]** contains: 0.00 **totalTransaction[2]** contains: 0.00 **totalTransaction[3]** contains: 0.00 **totalTransaction[19]** contains: 0.00			
	2	Date inputs encapsulates only one of the buyer's transactions	**Start date** input: 1/2/2024 **End date** input: 2/24/2024 **Date**: 4/7/2023 **transactions[0].buyerID** contains: 20(index 19 in users array) **transactions[0].transactAmount** contains: 2200.00	**totalTransaction[2]** contains: 11028.00	**totalTransaction[2]** contains: 11028.00	P

			Date: 2/22/2024 **transactions[1].buyerID** contains: 3 (index 2) **transactions[1].transactAmount** contains: 10588.00 **transactions[2].buyerID** contains: 3 (index 2) **transactions[2].transactAmount** contains: 440.00 **Date**: 5/5/2025 **transactions[1].buyerID** contains: 3 (index 2) **transactions[1].transactAmount** contains: 10588.00 **transactions[2].buyerID** contains: 3 (index 2) **transactions[2].transactAmount** contains: 440.00 **totalTransaction[2]** contains: 0.00			
	3	Date inputs encapsulates none of any buyer's transactions	**Start date** input: 1/2/2000 **End date** input: 2/24/2020 **Date:** 4/7/2023 **transactions[0].buyerID** contains: 20(index 19 in users array) **transactions[0].transactAmount** contains: 2200.00 **Date**: 2/22/2024 **transactions[1].buyerID** contains: 3 (index 2) **transactions[1].transactAmount** contains: 10588.00	Each element in **totalTransaction** array remains at 0.00	Each element in **totalTransaction** array remains at 0.00	P

			transactions[2].buyerID contains: 3 (index 2) **transactions[2].transactAmount** contains: 440.00 **Date**: 5/5/2025 **transactions[1].buyerID** contains: 3 (index 2) **transactions[1].transactAmount** contains: 10588.00 **transactions[2].buyerID** contains: 3 (index 2) **transactions[2].transactAmount** contains: 440.00 **totalTransaction[2]** contains: 0.00 **totalTransaction[19]** contains: 0.00			

Function	**#**	**Description**	**Sample Input Data**	**Expected Output**	**Actual Output**	**P/F**
AdminMenu	1	User tries to see all the registered users	**inputPassword** contains: H3LLo? **nChoice** contains: 1	showUsers gets called	showUsers gets called	P
	2	User tries to see all the registered sellers	**inputPassword** contains: H3LLo? **nChoice** contains: 2	showAllSellers gets called	showAllSellers gets called	P
	3	User tries to see total sales in a given duration	**inputPassword** contains: H3LLo? **nChoice** contains: 3	ShowTotalSales gets called	ShowTotalSales gets called	P

	4	User tries to see seller sales in a given duration	**inputPassword** contains: H3LLo? **nChoice** contains: 4	ShowSellersSales gets called	ShowSellersSales gets called	P
	5	User tries to see all the buyers in a given duration	**inputPassword** contains: H3LLo? **nChoice** contains: 5	ShowShopaholics gets called	ShowShopaholics gets called	P
	6	User tries to exit	**inputPassword** contains: H3LLo? **nChoice** contains: 6	User gets brought back to the main menu	User gets brought back to the main menu	P
	7	User inputs an invalid number	**inputPassword** contains: H3LLo? **nChoice** contains: 7	Program asks user for input again	Program asks user for input again	P
	8	User inputs the wrong admin password	**inputPassword** contains: Hello!	Prints the message “Wrong Password!” then the user gets brought back to main menu	Prints the message “Wrong Password!” then the user gets brought back to main menu	P

UNIV FUNC

Function	**#**	**Description**	**Sample Input Data**	**Expected Output**	**Actual Output**	**P/F**
getString	1	Single word name	**Input:** Prince	**string** contains: “Prince”	**string** contains: “Prince”	P
	2	Name with space in between	**Input:** Eula Lawrence	**string** contains: “Eula Lawrence”	**string** contains: “Eula Lawrence”	P
	3	Address with multiple spaces	**Input:** St Miguel Warp	**string** contains: “St	**string** contains: “St	P

			Zone	Miguel Warp Zone”	Miguel Warp Zone”	

Function	#	Description	Sample Input Data	Expected Output	Actual Output	P/F
swap	1	itemType a has greater productID than itemType b	**a.productID** contains: 5 **b.productID** contains: 4	**a.productID** contains: 4 **b.productID** contains: 5	**a.productID** contains: 4 **b.productID** contains: 5	P
	2	itemType b has greater productID than itemType a	**a.productID** contains: 5 **b.productID** contains: 6	**a.productID** contains: 5 **b.productID** contains: 6	**a.productID** contains: 5 **b.productID** contains: 6	P
	3	itemType a has the same productID as itemType b	**a.productID** contains: 5 **b.productID** contains: 5	**a.productID** contains: 5 **b.productID** contains: 5	**a.productID** contains: 5 **b.productID** contains: 5	P

Function	#	Description	Sample Input Data	Expected Output	Actual Output	P/F
sortproductID	1	items' productIDs are unsorted	**items[0].productID** contains: 5 **items[1].productID** contains: 3 **items[2].productID** contains: 2	**items[0].productID** contains: 2 **items[1].productID** contains: 3 **items[2].productID** contains: 5	**items[0].productID** contains: 2 **items[1].productID** contains: 3 **items[2].productID** contains: 5	P
	2	items' productIDs are unsorted, first value is lowest	**items[6].productID** contains: 7 **items[7].productID** contains: 9 **items[8].productID** contains: 8	**items[6].productID** contains: 7 **items[7].productID** contains: 8 **items[8].productID** contains: 9	**items[6].productID** contains: 7 **items[7].productID** contains: 8 **items[8].productID** contains: 9	P
	3	items' productID's are sorted in increasing order	**items[0].productID** contains: 1 **items[1].productID** contains: 2 **items[2].productID** contains: 3	**items[0].productID** contains: 1 **items[1].productID** contains: 2 **items[2].productID** contains: 3	**items[0].productID** contains: 1 **items[1].productID** contains: 2 **items[2].productID** contains: 3	P

Function	#	Description	Sample Input Data	Expected Output	Actual Output	P/F
count	1	itemType items[]	**items[0].seller**	**count**	**count**	P

		has 6 items that has the sellerID of 1	**ID** up until **items[5].sellerID** contains: 1 **person.userID** contains: 1	contains: 6	contains: 6	
	2	itemType items[] has 3 items that has the sellerID of 2	**items[6].sellerID** = 2 **items[7].sellerID** = 2 **person.userID** contains: 2	**count** contains: 2	**count** contains: 2	P
	3	itemType items[] has 1 items that has the sellerID of 3	**items[8].sellerID** = 3 **person.userID** contains: 3	**count** contains: 1	**count** contains: 1	P

Function	#	Description	Sample Input Data	Expected Output	Actual Output	P/F
sortUserID	1	OtherPeople[]'s userID are unsorted	**otherpeople[0].userID** contains: 3 **otherpeople[1].userID** contains:5 **otherpeople[2].userID** contains: 2	**otherpeople[0].userID** contains: 2 **otherpeople[1].userID** contains:3 **otherpeople[2].userID** contains: 5	**otherpeople[0].userID** contains: 2 **otherpeople[1].userID** contains:3 **otherpeople[2].userID** contains: 5	P
	2	Otherpeople[]'s userIDs are sorted in increasing order	**otherpeople[3].userID** contains: 7 **otherpeople[4].userID** contains:8 **otherpeople[5].userID** contains: 9	**otherpeople[3].userID** contains: 7 **otherpeople[4].userID** contains:8 **otherpeople[5].userID** contains: 9	**otherpeople[3].userID** contains: 7 **otherpeople[4].userID** contains:8 **otherpeople[5].userID** contains: 9	P
	3	Otherpeople[]'s userIDs are unsorted, last value is largest	**otherpeople[3].userID** contains: 9 **otherpeople[4].userID** contains:8 **otherpeople[5].userID** contains: 10	**otherpeople[3].userID** contains: 8 **otherpeople[4].userID** contains: 9 **otherpeople[5].userID** contains: 10	**otherpeople[3].userID** contains: 8 **otherpeople[4].userID** contains: 9 **otherpeople[5].userID** contains: 10	P

Function	#	Description	Sample Input Data	Expected Output	Actual Output	P/F
LoadUsers	1	Users.txt does not exist yet	N/A	Prints the message	Prints the message	P

				“Users.txt does not exist”	“Users.txt does not exist”	
	2	Users.txt exist	N/A	**users[]** and ***numberofusers** gets updated	**users[]** and ***numberofusers** gets updated	P
	3	File contains data	Users.txt contains: 5 Secretpass Chester Mondstadt City 1101011	**users[0].userID** contains: 5 **Users[0].password** contains: Secretpass **users[0].userName** contains: Chester **Users[0].address** contains: Mondstadt City **users[0].contactNo** contain: 1101011	**users[0].userID** contains: 5 **Users[0].password** contains: Secretpass **users[0].userName** contains: Chester **Users[0].address** contains: Mondstadt City **users[0].contactNo** contain: 1101011	P

Function	**#**	**Description**	**Sample Input Data**	**Expected Output**	**Actual Output**	**P/F**
LoadItems	1	Items.txt does not exist yet	N/A	Prints the message “Items.txt does not exist”	Prints the message “Items.txt does not exist”	P
	2	Items.txt exists	N/A	**Items[]** and ***itemNo** gets updated based on the contents of Items.txt	**Items[]** and ***itemNo** gets updated based on the contents of Items.txt	P
	3	File contains data	Items.txt contains: 10 4 PlusCoupon Hopium Adds points to grade 20 999	**Items[0].productID** contains: 10 **Items[0].sellerID** contains: 4 **Items[0].itemName** contains: PlusCoupon **Items[0].itemCategory** contains: Hopium **Items[0].itemDesc** contains: Adds points to grade	**Items[0].productID** contains: 10 **Items[0].sellerID** contains: 4 **Items[0].itemName** contains: PlusCoupon **Items[0].itemCategory** contains: Hopium **Items[0].itemDesc** contains: Adds points to grade	P

				Items[0].itemQty contains: 20 **Items[0].itemPrice** contains: 999	**Items[0].itemQty** contains: 20 **Items[0].itemPrice** contains: 999	

Function	**#**	**Description**	**Sample Input Data**	**Expected Output**	**Actual Output**	**P/F**
SaveUsers	1	users[] does not contain any data	N/A	Users.txt remains empty	Users.txt remains empty	P
	2	users[0] contains data	**users[0].userID** contains: 5 **Users[0].password** contains: Secretpass **users[0].userName** contains: Chester **Users[0].address** contains: Mondstadt City **users[0].contactNo** contain: 1101011	Users.txt contains: 5 Secretpass Chester Mondstadt City 1101011	Users.txt contains: 5 Secretpass Chester Mondstadt City 1101011	P
	3	users[0] and users[1] contains data	**users[0].userID** contains: 5 **Users[0].password** contains: Secretpass **users[0].userName** contains: Chester **Users[0].address** contains: Mondstadt City **users[0].contactNo** contain: 1101011 **users[1].userID** contains: 6 **Users[1].password** contains: realpass **users[1].userName** contains: Stan **Users[1].address** contains: Apple Street **users[1].contactNo** contain: 88888	Users.txt contains: 5 Secretpass Chester Mondstadt City 1101011 6 realpass Stan Apple Street 88888	Users.txt contains: 5 Secretpass Chester Mondstadt City 1101011 6 realpass Stan Apple Street 88888	P

Function	**#**	**Description**	**Sample Input Data**	**Expected Output**	**Actual Output**	**P/F**
SaveItems	1	items[] does not contain any data	N/A	Items.txt remains empty	Items.txt remains empty	P
	2	items[0] contains data	**Items[0].productID** contains: 10 **Items[0].sellerID** contains: 4 **Items[0].itemName** contains: PlusCoupon **Items[0].itemCategory** contains: Hopium **Items[0].itemDesc** contains: Adds points to grade **Items[0].itemQty** contains: 20 **Items[0].itemPrice** contains: 999	Items.txt contains: 10 4 PlusCoupon Hopium Adds points to grade 20 999	Items.txt contains: 10 4 PlusCoupon Hopium Adds points to grade 20 999	P
	3	items[0] and items[1] contains data	**Items[0].productID** contains: 10 **Items[0].sellerID** contains: 4 **Items[0].itemName** contains: PlusCoupon **Items[0].itemCategory** contains: Hopium **Items[0].itemDesc** contains: Adds points to grade **Items[0].itemQty** contains: 20 **Items[0].itemPrice** contains: 999 **Items[1].productID** contains: 15 **Items[1].sellerID** contains: 6 **Items[1].itemName** contains: Apple **Items[1].itemCategory** contains: Food **Items[1].itemDesc** contains: an apple a day.. **Items[1].itemQty** contains: 100 **Items[1].itemPrice** contains: 20	Items.txt contains: 10 4 PlusCoupon Hopium Adds points to grade 20 999 15 6 Apple Food an apple a day.. 100 20	Items.txt contains: 10 4 PlusCoupon Hopium Adds points to grade 20 999 15 6 Apple Food an apple a day.. 100 20	P