

Function	#	Description	Sample Input Data	Expected Output	Actual Output	P/F
getString	1	doesn't stop taking input when space is encountered	Seller 1	Seller 1	Seller 1	P
registerUser	1	checks if inputted userID is unique	userID inputted is 111	userID is 111	userID is 0	F
	2	^fixed, wrong algorithm	userID inputted is 111	userID is 111	userID is 111	P
loginUser	1	use valid login details	login details are correct	successful login	successful login	P
	2	use invalid login details	login details are wrong	unsuccessful login	unsuccessful login	P
findUser	1	index is correct	details of first user in user[] array	return 0	return 0	P
countItems	1	count is correct	user with 0 items	return 0	return 0	P
	2	count is correct	user with 1 item	return 1	return 1	P
sellAddItem	1	check if inputted productID is unique	unique ID 555 inputted	productID is 555	productID is 0	F
	2	^fixed, wrong algorithm	unique ID 555 inputted	productID is 555	productID is 555	P
	3	input taken ID	ID 555 inputted again	error case	program proceeded	F
	4	^fixed algorithm to only ask for other details if ID is unique	ID 555 inputted again	program won't ask for other details	program won't ask for other details	P
swapItems	1	function properly swaps	item[0] and item[1]	item[1] takes item[0], item[0] takes item[1]	item[1] takes item[0], item[0] takes item[1]	P
sellShowMyProducts	1	function properly sorts products based on productID	item[0] has productID 555 item[1] has productID 123	item[1] takes item[0], item[0] takes item[1]	item[1] takes item[0], item[0] takes item[1]	P
sellEditStock	1	all details of items are correctly edited	add 50 to item[0] quantity, which is currently 200	item[0] quantity = 250	item[0] quantity = 251	P
	2	all details of items are correctly edited	change item[0] price to 19.99	item[0] price = 19.99	item[0] price = 19.99	P
	3	all details of items are correctly edited	change item[0] name to "Corned Beef"	item[0] name = "Corned Beef"	item[0] name = "Corned Beef"	P
	4	all details of items are correctly edited	change item[0] category to "Canned Food"	item[0] category = "Canned Food"	item[0] category = "Canned Food"	P
	5	all details of items are correctly edited	change item[0] description to "A can of corned beef"	item[0] description = "A can of corned beef"	item[0] description = "A can of corned beef"	P
sellShowLowStock	1	only show items with quantity less than 5		show item[1], which has quantity of 3	show item[1], which has quantity of 3	P
swapUsers	1	function properly swaps	user[0] and user[1]	user[1] takes user[0], user[0] takes user[1]	user[1] takes user[0], user[0] takes user[1]	P
buyViewAll	1	sort users based on SellerID in increasing order	user[0] has ID 111, user[1] has ID 55	user[1] takes user[0], user[0] takes user[1]	user[1] takes user[0], user[0] takes user[1]	P
	1	correctly display products		correctly display products	display duplicated for each user	F
	2	^fixed, was iterating through number of items of the current user, instead of the number of users		correctly display products	correctly display products	P
	2	controls work properly		[N] shows the next seller	kept looping to the first seller when loop reached the last seller	F
	3	^fixed, note now shows up when iteration reached the last seller		[N] shows the next seller	[N] shows the next seller	P
buyShowSellerAll	1	correctly finds user and displays products in order		correct user and correct display	correct user and correct display	P
buyCategorySearch	1	correctly finds products of matching category and displays products with working controls	"Canned Food"	display "Corned Beef" and "Corned Tuna"	display "Corned Beef" and "Corned Tuna"	P
buyNameSearch	1	correctly finds products that contain the substring being searched and displays them with working controls	"Shirt"	display "Supreme Shirt" and "Sweatshirt"	display "Supreme Shirt" only	F
	2	^fixed, search is now case-insensitive	"Shirt"	display "Supreme Shirt" and "Sweatshirt"	display "Supreme Shirt" and "Sweatshirt"	P
	3	All caps input	"SHIRT"	display "Supreme Shirt" and "Sweatshirt"	display "Supreme Shirt" and "Sweatshirt"	P
						P
countCart	1	count is correct	cart with 0 items	return 0	return 0	P

	2	count is correct	cart with 2 items	return 2	return 2	P
buyEditCart	1	sorts user's cart based on productID in ascending order	cart[0] has ID 555 cart[1] has ID 123	cart[0] takes cart[1] cart[1] takes cart[0]	cart[0] takes cart[1] cart[1] takes cart[0]	P
	1	remove all items from seller	ID 111	cart[0] and cart[1] are removed	cart[0] and cart[1] are removed	P
	2	remove specific item	ID 123	cart[0] removed	cart[0] removed	P
	2	edit quantity	ID 555 quantity 5	cart[0] quantity = 5	cart[0] quantity = 6	P
countTransactions	1	count is correct	user with 0 transactions	return 0	return 0	P
	2	count is correct	user with 1 transaction	return 1	return 1	P
countTransactionItems	1	count is correct	transaction with 1 item	return 1	return 1	P
	2	count is correct	transaction with 2 items	return 2	return 2	P
buyCheckOut	1	correctly checks if quantity of item in cart changed	cart[0] quantity changed from 3 to 0, user was ordering 1	change quantity to 0	change quantity to 0	P
	2	correctly checks if price of item in cart changed	cart[0] price changed from 49999.99 to 39999.99	change price to 39999.99	change price to 39999.99	P
	3	correctly assign cart items to transaction	all	all details match	all details match	P
	3	display all transaction details correctly	all	all details match	Total Amount = 0 Seller ID = -1 Seller =	F
	4	^fixed, was removing items before showing final details	all	all details match	all details match	P
	4	write transaction data to Transaction.dat	specific item	changes reflect on the next program run	changes reflect on the next program run	P
	5	update product quantity in items array	specific item, quantity 5	item[0] quantity changes from 300 to 295	item[0] quantity changes from 300 to 295	P
	5	items removed from cart	specific seller	changes reflect	not removing from cart	F
	6	^fixed, was using the wrong algorithm and removing from transactions instead of cart	specific seller	changes reflect	changes reflect	P
adminSellers	1	displays users with items to sell		show only users with items in items[] array	show only users with items in items[] array	P
compareDates	1	earlier year than start year	06/08/2024 start date	invalid	invalid	P
current date: 04/03/2023		later year than end year	08/21/2022 end date	invalid	invalid	P
		same year, earlier month than start month	08/17/2023 start date	invalid	invalid	P
		same year, later month than end month	02/17/2023 end date	invalid	invalid	P
		same year, same month, earlier day than start day	04/17/2023 start date	invalid	invalid	P
		same year, same month, later day than end day	04/01/2023 end date	invalid	invalid	P
		valid date	04/01/2023 start date 04/30/2023 end date	valid	valid	P