Caleb Starkey

TextUI	calls
TextUI	None
MainMenu	None
RouteChoice	DisplayInventory(), AddItemToCart(), DisplayCart(), CartItems(), ItemMenu(), RemoveItemFromCart(), Checkout(), ClearCart.
ItemMenu	None

ShoppingCart	calls
ShoppingCart	None
AddItem	Calls get_quantity(), IncreaseQuantity(), and get_id().
RemoveItem	get_quantity(), and DecreaseQuantity().
DisplayCart	ToString().
ClearCart	None.
get_items()	None.

Item	calls
Item	None.
get_id()	None.
get_quantity()	None.
get_cost()	None.
get_type()	None.
IncreaseQuantity	None.
DecreaseQuantity	None.
ToString	None.
Clone	Item().
operator<<	None.

Store	Calls
Store	Item(), ShoppingCart().
DisplayInventory	ToString().
Items	ToString(), get_id(),get_quantity().
CartItems	get_items(), get_id(), ToString(), get_quantity().
AddItemToCart	get_id(), DecreaseQuantity(), Clone(), AddItem().
RemoveItemFromCart	get_items(), get_id(), RemoveItem(), IncreaseQuantity()
DisplayCart	DisplayCart().

Checkout	get_items(), get_cost(), get_quantity(), get_type(), ClearCart().
ClearCart	ClearCart().

Task 2: Clone initializes a new Item on the heap with the same field values as the item it's called on, and returns a pointer to the new Item. This is included because there are a lot of duplicate items being created to be added to the cart, thus it's far easier to have a method for cloning than include it in the addItemToCart method.

Task 3:

Class:	Cohesive:	Complete:	Clear:	Convenient:	Consistent:
Item	Yes, all methods have to do with the required information for an item in this implementation.	There are no setter methods which for this implementation isn't a huge problem but if this was a real store and price changes happened you'd have to change it in the store.txt file and restart the program.	Yes, it is clear what all methods/ fields are used for.	Yes, everything you could need to interact with the Item class in the rest of the implementation is provided	Getters, fields and other methods are nicely grouped by different but consistent naming structures.
Store	Yes, all methods in the class have to do with handling the store and the users cart. The implementation of a Shopper class could be	Yes, everything you would want to do within the store is implemented by a method.	Yes, it is clear what all the methods do and the purpose of the fields.	Yes, everything you could need to interact with the Store is included and easy to use.	Getters, fields and other methods are nicely grouped by different but consistent naming structures.

	argued for, but since one shopper at a time is what the implementation is for It's not really needed.				
ShoppingCart	Yes, all methods have to do with the shopping cart.	Yes, everything you need to implement a cart is included.	Yes it is clear what each method does.	Yes, all the methods you would want are included.	Getters, fields and other methods are nicely grouped by different but consistent naming structures.
TextUI	Yes, all methods and fields have to do with the handling of the menu/textUI	Yes, methods provide a complete handing of the main menu/textUI	Yes, it is very clear what each method does.	Yes, this class could potentially be handled in a main but this makes it far less cluttered and clear.	Getters, fields and other methods are nicely grouped by different but consistent naming structures.