Assignment 2

Goal:

The objective of this project was to create a shopping cart list that could handle mutations such as adding items, removing items, extending the length of the list, and shortening it. The user also should also be able to display the contents of the list. Since we know the size of the list and items will be mutable during runtime, this project will require a lot of dynamically allocated memory.

PseudoCode

- 1. Add Item
 - a. Get user inputs
 - i. Get name of item from user
 - ii. Get cost from user
 - iii. Get quantity from user
 - iv. Get unit from user
 - b. Build Item object
 - c. Check if item already exists in the list by comparing names *
 - i. If an object already exists,
 - 1. add to that object's quantity
 - 2. Delete new object
 - ii. If it does not exist
 - 1. Find the next NULL pointer in the list
 - 2. Assign the list pointer to the object
 - d. Calculate the total cost
- 2. Remove Item
 - a. Get user inputs
 - i. Get name of item from user
 - ii. Get number of items to remove
 - b. First try to find object in list by comparing names *
 - i. If there is a match
 - 1. Delete the item
 - 2. Maybe shorten the list
 - ii. If there is no match
 - 1. Do nothing
 - c. Calculate the total cost
- 3. Print List

- a. Iterate through the list and print off each items members
- b. Print out the number of items
- c. Print out the length of the list

List class

Private

- o Item ** List: This member will hold an array of Item pointers.
- Int num_items: the number of items that are in the list. Will be updated everytime and item is added or removed.
- Int size: The size of the array. Used to create a new array twice the size of the original when the array is completely filled, or to create an array half the size of the original if items only fill half the list.
- Int total_cost: The total cost of all the items in the list

Public:

- Setters and getters
- AddItem(): this function collects user's inputs to create an item, checks if that
 item already exists in the list, and then will either update the quantity of the item
 in the list if the item does exist, or will add the users item to the end of the list.
 Before adding the item, the function will determine if the list needs to be extended
 first or not. Finally, the total cost of all the items will be updated.
 - itemName(): This function gets the name of the item that the user entered and validates that it only contains letter characters and spaces.
 - maybeExtendList(): checks to see if the list is full of items. If so,
 - extendList(): Creates a new dynamically allocated array in memory that is twice the size of the current list
 - findMatch(): This function takes an Item-pointer parameter and compares its name member to each item in the list. If there is a match, the item in the list is returned. If there is no match, a NULL Item-pointer is returned.
 - In the *AddItem()* function, the item pointer parameter is initialized with the newly constructed item from the user's inputs
 - In the *RemoveItem()* function, a temporary item is created and is given the name of the item that the user wants to mutate. The temporary item initializes the parameter, and is deleted at the end of the function.
 - AddToQuantity(): this function takes an Item-pointer and an integer as parameters, and adds that integer to the item pointers quantity member. If the user enters an item that already exists in the list, this function updates that items quantity with the quantity that the user entered.
 - SubtractFromQuantity(): This function takes an item pointer and an integer as parameters and subtracts the quantity of that item by the integer.

- AddToEnd(): this function takes an item pointer parameter and adds it to the first NULL pointer in the list member. It is used if the user is adding an item to the list that does not already exist there.
- CalcTotalCost(): This function updates the total cost member by finding the sum of all the items external price members.
- removeltem(): This function will remove an item from the list, reduce the quantity
 of an item in the list, or do nothing and display a message if the item the user
 wants removed does not exist in the list.
- sort(): This function goes through the list member looking for NULL pointers that have items one iteration forwards. This is likely to be the case after an item is removed from the list. After this function is finished, all the NULL pointers should be grouped together at the end of the array (from using the swapItems() function).
- o **swapItems():** Switches an item in the list with another item.
- MaybeShortenList(): This function checks if the quotient between the number of items in the list and the size of the list equals two. It so, then a new array half the size of the original is created and replaces the original list.
- ShortenList(): Replaces the current array with a new array that is half the size of the original.

Item class

- Private:
 - String name: the name of the item
 - Unit unit: the items units
 - Int quantity: the number of items
 - Double unit_price: the price of the unit
 - Double ext price: the total price of that item (quantity * price);

Public:

- Setters and getters
- Operator==(const Item &obj): this function compares two items names. If they
 are the same, returns true. Else, it returns false.
- calcExtPrice(): This function multiplies the quantity of the item with the unit price member and sets the ext price member with the product.
- Item constructor(name = "", quantity = 0; unit_price = 0.00, Unit = CAN): The
 default values are used as fillers in the temporary object that is created when the
 user goes to remove an item.

Tests:

Adding an Item The user can add an item successfully. Iname = apple, quantity = 1 Item will be created with mer Y The user can add an item successfully. Item will be created with mer Item some and the same name as the first item in the list Test adding and Item that already exists in the list Total cost changes accordingly Removing an Item If the user tries to remove an item from the list that does not exist, nothing will happen If the user tries to remove an item successfully Enter the name of an item that exists in the list In the list User can remove an item successfully User can reduce the quantity of an item Item Print Print Print Print Print function display each items to the list, and size of the list.	Tests	Input	Expected Output	Pass (y/n)	Comments
The user can add an item successfully. The user can add multiple items to the list Test adding and Item that already exists in the list Total cost changes accordingly Removing an Item User can remove an item successfully User can remove an item successfully User can reduce the quantity of an item User can reduce the quantity of an item Print Print Print Print Print The user can add multiple items to the list different names illist and is asked how many items to remove. The user enters the highest number of items in the list, and since with the same name as the first item in the list and is asked how many items to remove. The user enters the list and is asked how many items to remove. The user enters the list, and since with a list of all the items with the same than alite mith the same of an item that is in the list and is asked how many items to remove. The user enters the name of an item in the list, and so well as the total cost, number of items in the list, and so well as the total cost, number of items in the list, and so well as the total cost, number of items in the list, and so well as the total cost, number of items in the list, and so well as the total cost, number of items in the list, and so well as the total cost, number of items in the list, and so well as the total cost, number of items in the list, and so well as the total cost, number of items in the list, and so well as the total cost, number of items in the list, and so well as the total cost, number of items in the list, and so well as the total cost, number of items with the immens displayle and item with the items with the items with the tems with the tems with the user of all the items with the new original quantity. Y If the user enters two items with the same name but with different prices, the calculated extendable price will reflect to show the sum of the original quantity. Y If the user enters two items with the same name but with		mput	Expedica Output	1 455 (9/11)	Comments
The user can add multiple items to the list Test adding and Item that already exists in the list Total cost changes accordingly Removing an Item If the user tries to remove an item from the list that does not exist, nothing will happen User can remove an item successfully User can reduce the quantity of an item User can reduce the quantity of an item Print Print Print Print Test adding and Item that already exists in the list dad an item with the same name st the first item in the list and lis asked how many items to remove. The user enters the highest number. Add an item with the same name she thirth the same name as the first item in the list original quanity and the new quantity. For each item added, the total cost reflects the sum of all the item extended price If the user enters two items with the same name but with different prices, the calculated extendable price will reflect the price of the original item. For each item dadd, the original quanity and the new quantity of an item that exists in the list and is asked how many items to remove. The user enters the highest number. Print Print Print function display each items name, quantity, units, price, and extended price, as well as the total cost, number of items in the list, and cost and number. Alda nitem with the same name with the flath elems with their and the unamity should change to show the sum of the quantity, should change to show the sum of the quantity should change to show the sum of the quantity should change to show the sum of the quantity should change to show the sum of the quantity should change to show the sum of the quantity should change to show the sum of the quantity should change to show the sum of the quantity should change to show the sum of the quantity should change to show the		(name - apple guantity - 1	Itam will be areated with more	V	
list different names Test adding and Item that already exists in the list Total cost changes accordingly Add some items to the list Total cost changes accordingly Removing an Item If the user tries to remove an item from the list that doesn't exist in the list If the user name but with different prices, the original quanity and the new quantity. For each item added, the total cost reflects the sum of all the item extended price If the user extended price will reflect the price of the original item. For each item added, the total cost reflects the sum of all the item extended price If the user enters two items with the same name but with different prices, the calculated extendable price will reflect the price of the original item. For each item added, the total cost reflects the sum of all the item extended price If the user enters two items with the same name but with different prices, the original quanity and the new quantity. For each item added, the total cost reflects the sum of all the item extended price For each item added, the total cost reflects the sum of all the item extended price For each item added, the total cost reflects the sum of all the item could not be found There was a segmentation fault. UPDATE: a new approach for deleting the items was used that was successful. This can be read in the reflections section of the design document. The quantity member of the item in the list should reflect correct difference The quantity member of the item in the list should reflect correct difference The quantity member of the item in the list should reflect correct difference The quantity member of the item in the list should reflect correct difference The quantity member of the item in the list should reflect correct difference The quantity member of the item in the list should reflect to the price of the original quantity and the new original quantity and the	•			T	
Test adding and Item that already exists in the list Add an item with the same name as the first item in the list Total cost changes accordingly Add some items to the list Total cost changes accordingly Add some items to the list For each item added, the total cost reflects the sum of all the item extended price Removing an Item If the user tries to remove an item from the list that does not exist, nothing will happen Enter the name of an item that doesn't exist in the list in the list Item should be removed with no segmentation faults User can remove an item successfully User can reduce the quantity of an item that is in the list and is asked how many items to remove. The user enters the highest number. Print Print Print Print function display each items and extended price, as well as the total cost, number of items in the list, and should be removed with program should be removed with in the list should reflect correct difference The quantity should change to show the sum of the original item. The quantity should change to show the sum of the original quantity and the new quantity, and the new quantity of an item added, the total cost reflects the sum of all the item extended price and the price of the original item. The quantity should change to show the new quantity and the new quantity and the new calculated extendable price, will the item in the list and item could not be found There was a segmentation fault. UPDATE: a new approach for deleting the items was used that was successful. This can be read in the reflections section of the item in the list should reflect correct difference The quantity member of the item in the list should reflect correct difference The quantity member of the item in the list should reflect correct difference Y There was a segmentation fault. UPDATE: a new approach for deleting the items was used that was successful. This can be read in the reflections section of the item in the list should reflect correct difference The quantity and the new calculated	4			Υ	
Test adding and Item that already exists in the list Add an item with the same name as the first item in the list Total cost changes accordingly Removing an Item If the user tries to remove an item from the list that does not exist, nothing will happen User can remove an item successfully User can reduce the quantity of an item I user can reduce the quantity of an item I we can reduce the quantity of an item I we can reduce the quantity of an item I we can reduce the quantity of an item I we can reduce the quantity of an item I we can reduce the quantity of an item I we can reduce the quantity of an item I we can reduce the quantity of an item I we can reduce the quantity of an item I we can reduce the quantity of an item I we can reduce the quantity of an item I we can reduce the quantity of an item that is in the list and is asked how many items to remove. The user enters the highest number. I we can reduce the quantity of an item that is in the list and is asked how many items to remove. The user enters the highest number. I we can reduce the quantity of an item that is in the list and is asked how many items to remove. The user enters the highest number. I we can reduce the quantity of an item that is in the list and is asked how many items to remove. The user enters the highest number. I we can reduce the quantity of an item that exists in the list and is asked how many items to remove. The user enters the highest number. I we can reduce the quantity of an item that is in the list and is asked how many items to remove. The user enters the highest number. I we can reduce the quantity of an item that is in the list and is asked how many items to remove. The user enters the highest number. I we same name but with different exident, the calculated extendable price will reflect the total cost reflects the sum of all the total cost reflects the sum of all the title total cost reflects the sum of all the title total cost reflects the sum of all the title total cost reflects the sum of all	list	different names			
exists in the list name as the first item in the list original quanity and the new quantity Total cost changes accordingly Removing an Item If the user tries to remove an item from the list that does not exist, nothing will happen User can remove an item successfully User can reduce the quantity of an item that is in the list in the list in the user tries to remove. The user enters the highest number. Print Print Print function display each items a the first item in the list original quantity and the new quantity of all the new quantity and the new total cost reflects the sum of all the item and the liet and is all the item and the item could not be found There was a segmentation fault. UPDATE: a new approach for deleting the items was used that was successful. This can be read in the reflections section of the item in the list should reflect correct difference The quantity and the new quantity and the new calculated. The program should let uesr know that the item could not be found There was a segmentation fault. UPDATE: a new approach for deleting the items was used that was successful. This can be read in the reflect to the price of the original item. The quantity and the new calculations and the item could not be found that the ite		name as the first item in		Y	
exists in the list the list original quanity and the new quantity quantity For each item added, the total cost reflects the sum of all the item extended price Removing an Item If the user tries to remove an item from the list that does not exist, nothing will happen Enter the name of an item that doesn't exist in the list name of an item that exists in the list user can remove an item successfully User can reduce the quantity of an item that is in the list and is asked how many items to remove. The user enters the highest number. Print Print Print Print function display each items name, quantity, units, price, and extended price, as well as the total cost, number of items in the list, and original quantity with and quantity. The quantity and the new quantity. For each item added, the total cost reflects the sum of all the item added, the total cost reflects the sum of all the item extended price. Print Print None original quantity and the new quantity and the new quantity and the new quantity. The reach item added, the total cost reflects the sum of all the item added, the total cost reflects the sum of all the item extended price. Print None original quantity and the new quantity and the new quantity. The quantity and the new cache item added, the total cost reflects the sum of all the item extended price. Y There was a segmentation fault. UPDATE: a new approach for deleting the items was used that was successful. This can be read in the reflections section of the item in the list should reflect to correct difference Y The quantity member of the item could not be found The quantity member of the item in the list should reflect to correct difference Y Difference The quantity and the new cache item should be removed with no segmentation fault. The quantity and the new cache item sould not be found The quantity and the reflect to the total total the item could not be found The quantity and the reflect total the total total the item could not be found The quantity and the reflect					
Total cost changes accordingly Add some items to the list For each item added, the total cost reflects the sum of all the item extended price Removing an Item If the user tries to remove an item from the list that does not exist, nothing will happen Enter the name of an item that doesn't exist in the list In the list In the list User can remove an item successfully User can reduce the quantity of an item Item User can reduce the quantity of an item Item Print Print Print function display each items name, quantity, units, price, and extended price, as well as the total cost, number of items in the list, and Add some items to the list For each item added, the total clost reflects the sum of all then total cost, reflects the sum of all the total cost, reflects the sum of all the item extended price Print Print function display each items name, quantity, units, price, and extended price, as well as the total cost, number of items in the list, and			original quanity and the new		·
Total cost changes accordingly Removing an Item If the user tries to remove an item from the list that does not exist, nothing will happen User can remove an item successfully User can reduce the quantity of an item that is in the list or move. The user enters the highest number. Print Print function display each items name, quantity, units, price, and extended price, as well as the total cost, reflects the sum of all the item extended price all the item extended price all the item extended price and the item extended price all the item extended price and item that doesn't exist in the list obe found There was a segmentation fault. UPDATE: a new approach for deleting the items was used that was successful. This can be read in the reflections section of the item in the list should reflect correct difference The quantity member of the item in the list should reflect correct difference Y In the list and item that exists in the list and item that is in the list and item that exists in the list and item should be removed with no segmentation faults The quantity member of the item in the list should reflect correct difference Y There was a segmentation fault. UPDATE: a new approach for deleting the items was used that was successful. This can be read in the reflections section of the item in the list should be removed with no segmentation faults The quantity member of the item in the list should reflect correct difference Y			quantity		the price of the original item.
Removing an Item If the user tries to remove an item from the list that does not exist, nothing will happen User can remove an item successfully User can reduce the quantity of an item item If the user tries to remove an item successfully User can reduce the quantity of an item that is in the list in the list in the list in the list asked how many items to remove. The user enters the highest number. Print Print Print function display each items name, quantity, units, price, and extended price, as well as the total cost, number of items in the list, and All the item extended price and item should let uesr know that the item could not be found There was a segmentation fault. UPDATE: a new approach for deleting the items was used that was successful. This can be read in the reflections section of the design document. The quantity member of the item in the list should reflect correct difference Y There was a segmentation fault. UPDATE: a new approach for deleting the items was used that was successful. This can be read in the reflections section of the design document. The quantity member of the item in the list should reflect correct difference Y There was a segmentation fault. UPDATE: a new approach for deleting the items was used that was successful. This can be read in the reflections section of the design document. The quantity member of the item in the list should reflect correct difference Y There was a segmentation fault. UPDATE: a new approach for deleting the items was used that was successful. This can be read in the reflections section of the design document. The quantity member of the item in the list should reflect correct difference Y There was a segmentation fault. UPDATE: a new approach for deleting the items was used that was successful. This can be read in the reflections section of the design document.	Total cost changes accordingly	Add some items to the list	For each item added, the	Y	
Removing an Item If the user tries to remove an item from the list that does not exist, nothing will happen User can remove an item successfully User can reduce the quantity of an item that is in the list item should be removed with item should be removed a litem that is in the list and litem should be removed with no segmentation faults User can reduce the quantity of an item that is in the list and litem in the list should reflect correct difference Print Print function display each items name, quantity, units, price, and extended price, as well as the total cost, number of items in the list, and			total cost reflects the sum of		
If the user tries to remove an item from the list that does not exist, nothing will happen Enter the name of an item that doesn't exist in the list User can remove an item successfully User can reduce the quantity of an item item User can reduce the quantity of an item Item User can reduce the quantity of an item that is in the list to remove. The user enters to remove. The user enters name, quantity, units, price, and extended price, as well as the total cost, number of items in the list, and			all the item extended price		
from the list that does not exist, nothing will happen User can remove an item successfully User can reduce the quantity of an item that is in the list User can reduce the quantity of an item that is in the list to remove. The user enters to remove. The user enters the highest number. Print Print function display each items name, quantity, units, price, and extended price, as well as the total cost, number of items in the list, and Enter the name of an item that is in the list whow that the item could not be found There was a segmentation fault. UPDATE: a new approach for deleting the items was used that was successful. This can be read in the reflections section of the item in the list should reflect correct difference The quantity member of the item in the list should reflect correct difference Y There was a segmentation fault. UPDATE: a new approach for deleting the items was used that was successful. This can be read in the reflections section of the design document. Y Print function display each items name, quantity, units, price, and extended price, as well as the total cost, number of items in the list, and	Removing an Item				
that doesn't exist in the list nothing will happen User can remove an item successfully User can reduce the quantity of an item User can reduce the quantity of an item Print Print Print function display each items name, quantity, units, price, and extended price, as well as the total cost, number of items in the list, and know that the item could not be found N There was a segmentation fault. UPDATE: a new approach for deleting the items was used that was successful. This can be read in the reflections section of the item in the list should reflect correct difference Y The quantity member of the item in the list should reflect correct difference Y The quantity member of the item in the list should reflect correct difference Y Definit None No	If the user tries to remove an item		program should let uesr		
User can remove an item successfully User can reduce the quantity of an item that is in the list asked how many items to remove. The user enters the highest number. Print Print Print function display each items name, quantity, units, price, and extended price, as well as the total cost, number of items in the list, and Defound Item should be removed with no segmentation faults Item should be removed with no segmentation faults N The quantity member of the item in the list should reflect correct difference Y The quantity member of the item in the list should reflect correct difference Y The quantity member of the item in the list should reflect correct difference Y The quantity member of the item in the list should reflect correct difference Y The quantity member of the item in the list should reflect correct difference Y The quantity member of the item in the list should reflect correct difference Y The quantity member of the item in the list should reflect correct difference Y The quantity member of the item in the list should reflect correct difference Y The quantity member of the item in the list should reflect correct difference Y The quantity member of the item in the list should reflect correct difference Y The quantity member of the item in the list should reflect correct difference Y	from the list that does not exist,		know that the item could not	Υ	
User can remove an item successfully User can remove an item successfully User can reduce the quantity of an item that is in the list and is asked how many items to remove. The user enters the highest number. Print Print function display each items name, quantity, units, price, and extended price, as well as the total cost, number of items in the list, and The quantity member of the item in the list should reflect correct difference The quantity member of the item in the list should reflect correct difference Y There was a segmentation fault. UPDATE: a new approach for deleting the items was used that was successful. This can be read in the reflections section of the design document. The quantity member of the item in the list should reflect correct difference Y ber appropriately with zero se Y There was a segmentation fault. UPDATE: a new approach for deleting the items was used that was successful. This can be read in the reflections section of the design document. Y ber appropriately with zero se Y	nothing will happen	that doesn't exist in the list	be found		
User can reduce the quantity of an item that is in the list and is asked how many items to remove. The user enters the highest number. Print Print function display each items name, quantity, units, price, and extended price, as well as the total cost, number of items in the list, and None item that is in the list and is asked how many items item in the list should reflect correct difference Y The quantity member of the item in the list should reflect correct difference Y The quantity member of the item in the list should reflect correct difference Y Description The quantity member of the item in the list should reflect correct difference Y Description The quantity member of the item in the list should reflect correct difference Y Description The quantity member of the item in the list should reflect correct difference Y Description The quantity member of the item in the list should reflect correct difference Y Description The quantity member of the item in the list should reflect correct difference Y Description The quantity member of the item in the list should reflect correct difference Y Description The quantity member of the item in the list should reflect correct difference Y Description The quantity member of the item in the list should reflect correct difference Y Description The quantity member of the item in the list should reflect correct difference Y Description The quantity member of the item in the list should reflect correct difference Y Description The quantity member of the item in the list should reflect correct difference Y Description The quantity member of the item in the list should reflect correct difference Y Description The quantity member of the item in the list should reflect correct difference Y	User can remove an item successfully	in the list		N	a new approach for deleting the items was used that was successful. This can be read in the reflections section of the
Print function display each items name, quantity, units, price, and extended price, as well as the total cost, number of items in the list, and		item that is in the list and is asked how many items to remove. The user enters	item in the list should reflect	Y	
name, quantity, units, price, and extended price, as well as the total cost, number of items in the list, and	Print				
extended price, as well as the total cost, number of items in the list, and	Print function display each items				
cost, number of items in the list, and	name, quantity, units, price, and				
	extended price, as well as the total	None	ber appropriately with zero se	Y	
size of the list	cost, number of items in the list, and				
	size of the list				

Reflection:

In my original design, I decided that my add item and remove item functions would both need to traverse the list to see if an item exists in the list. If a match was found, the function would return the item (Item *). The add and remove functions might then mutate the quantity of the item, or the remove function could delete the dynamic memory. The former cases passed their case tests; the latter however caused a memory leak.

The memory leak could be because the "find item" function initialized another pointer to the same item in a different scope, which was used to delete the item if the items quantity reached zero. The list still contained a pointer to that allocated object and caused bugs when that pointer was referenced in other places in the program (i.e. the print function). To resolve, I created a separate function to delete items directly from the list. I made sure that my

segmentation faults were gone by running by executable with valgrind a few times. The lesson here is to be careful when creating multiple pointers to an object to try to avoid leaving dangling pointers.

New Functions:

- **Deletion()**: This function takes an item pointer parameter, finds the item in the list, deletes the item, and sets the item pointer to NULL. This function was created to resolve a segmentation fault as mentioned in my reflection section.
- **DeleteItem()**: This function takes an item pointer as a parameter and compares the value of that pointer to each item in the list. If a match is found, the item in the list is deleted and replaced with a NULL value.