**Struct ProductList**

declare itemNum;

declare itemName;

declare itemCost;

declare itemQty;

**Struct ShoppingCart**

Declare itemName;

Declare itemQty;

**Function main**

***Input:*** fileName, inFile, productPurchased, prevStored, newQty, sizeProductData, productExists, rowFound, customerInfo[5], \*product, \*cart

***Processing:*** count, newQty

***Output:*** N/A

***Algorithm***

1. Open fileName
2. If filename opens

Display “If you would like to continue to checkout, enter 0.”

Display “If you would like to view your cart, enter cart.”

Set \*product to function readFile()

Declare \*cart according to sizeProductData

For (count=0, count < sizeProductData , count++)

Set cart[count].itemName to function productPurchase

Check if product has already been added to shopping cart

for (int stored=0, stored < sizeProductData, stored++)

If cart[stored].itemName == cart[count].itemName

productPurchased=true, prevStored=stored

If user enters “cart”

Call displayCart()

If cart[count].itemName=0

Display “name”

Get customerInfo[0]

Display “street address”

Get customerInfo[1]

Display “city”

Get customerInfo[2]

Display “state”

Get customerInfo[3]

Display “zip code”

Get customerInfo[4]

Call displayInvoice()

Call updateInventory()

Break loop

Set productExists to function searchProduct

If productExists = false

Output “product does not exist”

Else If productExists = true

If productPurchased=true

Output “how many of the item would you like”

Store answer in newQty

Set newQty to newQty+product quanityt previously purchased

Set cart.[prevStored].itemQty to updated quantity

While function quanitityProduct =false

Display “reenter quantity”

Set cart[prevStored].itemQty

Else

Output “how many of the item would you like”

Set cart[count].itemQty to input from user

While function quanitityProduct =false

Display “reenter quantity”

Set cart[count].itemQty to input from user

**Function ProductList \*readFile:** reads file from productData.csv and stores the data int a struct variable

***Parameters:*** inFile by reference, sizeProductData by reference

***Processing:*** row

***Output:*** product

***Return type:*** pointer

***Algorithm***

1. Set the counter for row to 0
2. Store first line to sizeProductData
3. Declare \*product, a dynamic array
4. Do

Set productList[row].itemNum first item of file

Set productList[row].itemName second item of file

Set productList[row].itemCost third item of file

Set productList[row].itemQty fourth item of file

Increase row++

Set productList[row][1] first item of following line

While row < sizeProductData AND !end of file

1. Close file
2. Return product

**Function productPurchase:** asks the user which product they would like to purchase and return the product number

***Parameters****:* none

***Processing:*** productNumber

***Output:*** productNumber

***Return type:*** int

**Algorithm**

1. Get productNumber
2. Set productNumber
3. Return productNumber

**Function searchProduct:** searches for the product number in the array containing data from ProductData.csv

***Parameters: \****product, \*cart***,*** cartNumber***,*** rowFound by reference, sizeProductData by reference

***Processing:*** productExists

***Output:*** productExists

***Return type:*** Bool

***Algorithm***

1. For (row=0, row <145, row++)

If shoppingCart[cartNumber][row]!=productList[row][0]

Set productExists to false

Else

Set productExists to true

Output product Number and product Cost

Set row to rowFound

Return productExists

1. Return productExists

**Function quantatityProduct:** checks if the quantity being purchased is available and adds it to the shopping cart

***Parameters:***\*product, \*cart, cartNumber, rowFound by reference

***Processing:*** quantityExists

***Output:*** quantityExists

***Return type:*** Bool

***Algorithm***

1. If cart[rowFound].itemQty > product[rowFound].itemQty

Set quantityExists to false

Else

Set productExsists to ture

Display “Item has been added to shopping cart”

Change quanitity of product avilable

Return quanitityExists

1. Return quantityExists

**Function displayCart:** displays the current cart of the shopper

***Parameters:*** \*cart, cartProducts

***Processing:*** counter

***Output:*** n/a

***Return type***: void

***Algorithm:***

1. For(int counter=0, counter < cartProducts; counter++)

Display “product name” and cart[counter].itemName

Display “quantity purchasing” and cart[counter].itemQty

**Function displayInvoice:** displays the final invoice statement to a text file and the console

***Parameters:*** customerInfo[5], \*product, \*cart, cartNumber

**Processing:** counter, row, outFile, total

***Output:*** int

***Return type:*** int

***Algorithm***

1. Open outFile “Invoice.txt”
2. If ofstream does not open

Return 0

Else

Output customerInfo to console

Display “item number, item name, qty, unit cost, total” to console

Output customerInfo to outFile

Outupue “item number, item name, qty, unit cost, total” to outFile

For (counter=0, counter< cartNum, counter++)

For (row=0, row< cartNum, row++)

If shoppingCart = productList

Set total to cart[i].itemQty\*product[row].itemCost

Output product[row].itemNum item number to console

Output cart[i].itemQty to console

Output unit cost= cart[i].itemQty\*product[row].itemCost

Output total to console

Set total to cart[i].itemQty\*product[row].itemCost

Output product[row].itemNum item number to outFile

Output cart[i].itemQty to outFile

Output unit cost= cart[i].itemQty\*product[row].itemCost

Output total to outFile

Output final total value to console

Output final total value to outFile

1. Return int

**Function updateInventory:** updates the ProductData.csv file with the inventory remaining after products have been purchased

***Parameters:*** outfile by reference, \*product, \*cart, sizeProductData by refernce, cartNum

**Processing:** counter1, counter2

***Output:*** n/a

***Return type:*** void

***Algorithm***

1. Opens productData.csv
2. If unable to open outputs error
3. Else

For (int counter1=0; counter1<cartNum; counter1++)

For (int counter2=0; counter2<sizeProductData; counter2++)

If (product[i].itemName==cart[r].itemName)

Find location of product line

Using seekp change the quanitity of ProductQuanity