# CIST2362 Software Design Document

Cash Register Program/Class object

Kimberly Jackson

# Introduction

A program is needed to allow for faster and easier checkout. Specifically, a Cash register is what will be simulated. A CashRegister class will be created to accept input from the cashier or consumer about what is being purchased. This information will link back into an InventoryItem class, which will supply information about the items being purchased to tally up the customer’s cost. Once all items have been entered, the purchase will be taxed and combined to complete the transaction. Of course, there will be a check to see if there are enough items on hand and subtract them from the supply in inventory. A 30% markup will be applied on all items to ensure a profit. The program will also display the subtotal, amount of sales tax, and the total of everything together.

Error checking will be in place to ensure that there are enough items in inventory and that a negative number is not input for number of items to be purchased.

# Use Cases

*Use Case: Charlie - single item - correctly*

* 1. *Charlie works as a cashier. A customer approaches with a single item.*
  2. *Charlie inputs the item’s code.*
  3. *Program asks how many of the item is being purchased.*
  4. *Charlie enters the number 1 and presses enter.*
  5. *The program checks to see that there is indeed one of the requested item in stock.*
  6. *Item is in stock. One is subtracted from the stock.*
  7. *Program asks if there are more items to be purchased.*
  8. *Charlie enters n for no.*
  9. *Program performs calculations and displays subtotal, tax, and total.*
  10. *Money exchanges hands.*
  11. *Program returns to idle screen.*

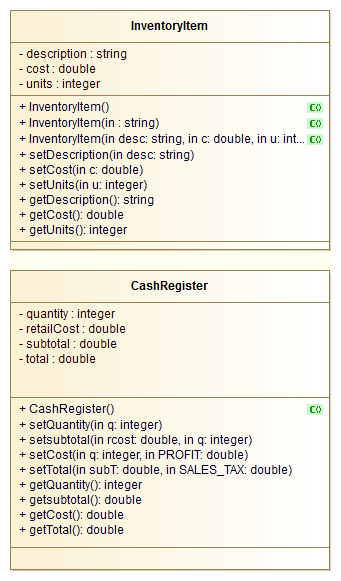
*Use Case: Charlie - multiple items - correctly*

1. *Charlie works as a cashier. A customer approaches with a basket of items.*
2. *Charlie inputs the first item’s code.*
3. *Program asks how many of the item is being purchased.*
4. *Charlie enters the number 3 and presses enter.*
5. *The program checks to see that there is enough of the requested item in stock.*
6. *Item is in stock. Three is subtracted from the stock.*
7. *Program asks if there are more items to be purchased.*
8. *Charlie enters y for yes.*
9. *Menu with all the items again. Charlie enters the item’s code.*
10. *Program asks how many of that item is being purchased.*
11. *Charlie enters 2 and presses enter.*
12. *Program checks current stock.*
13. *Number being purchased is subtracted from current stock.*
14. *Program performs calculations and displays subtotal, tax, and total including the previous item.*
15. *Program asks again if there are more items to purchase.*
16. *Charlie enters n for no.*
17. *Money exchanges hands.*
18. *Program returns to idle screen.*

*Use Case: Jenny - Error - Not enough stock*

1. *Jenny is checking out a customer with multiple items. She inputs the first item’s code.*
2. *The program asks how many of that item is being purchased.*
3. *Jenny enters 6 even though the customer has brought up the last 3 in stock.*
4. *The program checks that there are indeed 6 items of that code.*
5. *Program displays that there are only 3 in stock. Jenny is requested to enter a value less than the current stock.*
6. *Jenny keys in the proper number of items.*
7. *Program performs calculations and displays the subtotal, sales tax, and total.*
8. *Program asks if there are more items being purchased.*
9. *Process repeats without issue.*

# Design Overview



# System Tasks Description

*global constants double SALES\_TAX=.06, PROFIT = .3;*

## Function: main

*Create cashregister class*

*Call function with a greeting*

*while more stuff to buy = y*

*Call up the menu function*

*display do you have more stuff to buy?*

*end while*

## Function: menu

*const int SIZE*

*inventoryitem[SIZE]*

*for counter=0 to SIZE-1*

*display InventoryItem[counter]*

*end for*

*display “What would you like to purchase?”*

*input (options 1-SIZE)*

*switch:*

*case 1:*

*menuoption(inventoryitem[option-1])*

*break*

*case 2:*

*menuoption(inventoryitem[option-1])*

*break*

*etc with the cases*

*default*

*break*

*end function*

## Function: menuoption(inventoryitem intitem)

*how many of this would you like to purchase*

*input quantity*

*if quantity > 0*

*if quantity > item[option-1].getUnits*

*you done GOOFED, put in the proper number!*

*input quantity*

*else*

*double rcost += (inventoryitem[counter-1] \* PROFIT\_MARGIN)*

*cashregister.setsub(rcost)*

*cashregister.setTax(SALES\_TAX)*

*end if*

*else if quantity < 0*

*display cannot accept that number for the quantity to be purchased*

*input quantity*

*else*

*display oops I guess?*

*end if*

*end function*