SHOPPING CART PROJECT 3

Table of Contents

Statement of Independent Effort

I, JaKeyvan Jones, hereby certify that is my original work completed without the assistance of anyone or any outside resources.

JaKeyvan Jones

Analysis of Specification

Input

Input user_name, pass_word – “Username and password input by the user”

Input account_num, memberlvl – “Account number and membership level input by the user”

Input item_name, sku – “item name and stock keeping unit input by the user”

Input quan_tity – “User inputs the amount of the selected item he wants”

Input total- “ Amou

Output

Display "Enter username and password: " – “Asks user to enter username/password”

Display "Verify account by entering account number and member level: " – “Asks user to verify account by entering their account number and membership level”

Display "Access granted! Welcome back Jane: ” – “Allows login and displays welcome back message”

Display "What item would you like to purchase? Enter product name and stock keeping unit: " – "Asks user to enter product name and sku of item they would like to purchase"

Display prouducts.csv – "Displays all items the store has to offer"

Display "Error! Invalid choice! Product name and sku dont match: " – "Error message displayed when product name and stock keeping unit don't match"

Display "How many would you like to purchase? " – "Asks user how many of the selected item he/she would like to purchase"

Display "Your total is: " – "Displays total"

Display "Error! You do not have enough available funds to purchase items: " – "Error message displayed if there aren't enough funds on account to make purchase"

Display "Thank you for your purchase! Your updated account information is below: " – "Thank you message that lets you know your updated information will be listed below"

Display "Customer name- customer: " – "Displays customer's name"

Display "Username- user_name: " – "Displays customer's username"

Display "Password- pass_word: " – "Displays customer's password"

Display "Account number- account_num: " – “Displays customer’s account number”

Display "Member level- memberlvl: " – “Displays customer’s membership level”

Display "Store Credit- NewStoreCredit: " – “Displays customer’s new, updated account balance”

Assign

Assign user_name = jsmith, pass_word = blue123 – “Assigns username and password for login”

Assign account_num = 123456789, memberlvl = Gold – “Assigns account number and member level to verify account”

Assign total = (item_unit * quan_tity * item_price)(salestax) - percent_discount – “Assigns total equation”

Assign NewStoreCredit = account_bal – total – “Assigns NewStoreCredit equation”

Assign discountPercent = 0.12 - “Assigns discount percent to 12%”

Assign discountPercent = 0.085 - “Assigns discount percent to 8.5%”

Assign discountPercent = 0.06 - “Assigns discount percent to 6%”

Assign discountPercent = 0 - “Assigns discount percent to 0%”

If

if (username != user_name, password != pass_word) - "Beginning statement of what happens if username and password don't match the ones needed to login"

if (product name != item_name, product sku != sku) – "Beginning statement of what will happen if the product name and sku don't match"

if (total > account_bal) – "Beginning statement of what happens if the total is more than the available balance on the account"

if(total <= account_bal) – "Beginning statement of what happens if the total is less than or equal to the available balance on the account"

If memberLevel == "Diamond" "Begins statement on what happens if member has Diamond level account" If totalSpend > 700.00 - "Begins statement on what happens if user with Diamond level account spends at least $700"

Elseif totalSpend > 300.00 - "Begins statement on what happens if user with Diamond level account spends at least $300"

Elseif totalSpend > 100.00 - "Begins statement on what happens if user with Diamond level account spends at least $100"

Elseif memberLevel == "Gold" - "Begins statement on what happens if member has Gold level account"

If totalSpend > 300.00 “Begins statement on what happens if user with Gold level account spends at least $300”

Elseif totalSpend > 100.00 “Begins statement on what happens if user with Gold level account spends at least $100”

Elseif memberLevel == "Blue" - “Begins statement on what happens if member has Blue level account”

If totalSpend > 100.00 - “Begins statement on what happens if user with Blue level account spends at least $100”

Else totalSpend < 100.00 - “Begins statement on what happens if user spends less than a $100

Pseudocode

BEGIN

Declare string name, user_name, pass_word, add_ress, customer, username,

password, memberlvl, address

Declare int item_unit, quan_tity, account_num

Declare real item_price, account_bal, store_credit, salestax, discountPercent, total,

totalSpend;

Assign totalSpend = (item_price * item_unit * item_price)

ifstream myfile;

myfile.open(accounts.dat, products.csv);

Display "Enter username and password: "

Input user_name, pass_word

Assign user_name = jsmith, pass_word = blue123

Display "Verify account by entering account number and member level: "

Input account_num, memberlvl

Assign account_num = 123456789, memberlvl = Gold

Display "Access granted! Welcome back Jane: ”

if (username != user_name, password != pass_word){

myfile.close(accounts.dat, products.csv);

exit (EXIT_FAILURE);

Display "What item would you like to purchase? Enter product name and stock keeping unit: "

Display prouducts.csv

Input item_name, sku

if (product name != item_name, product sku != sku){

Display "Error! Invalid choice! Product name and sku dont match: "

Display "How many would you like to purchase? "

Input quan_tity

Display "Your total is: "

Assign total = (item_unit * quan_tity * item_price)(salestax) - discountPercent

If memberlvl == "Diamond"

If total > 700.00

Assign discountPercent = 0.12

Elseif totalSpend > 300.0

Assign discountPercent = 0.085

Elseif totalSpend > 100.00

Assign discountPercent = 0.06

Elseif memberLevel == "Gold"

If totalSpend > 300.0

Assign discountPercent = 0.085

Elseif totalSpend > 100.00

Assign discountPercent = 0.06

Elseif memberLevel == "Blue"

If totalSpend > 100.00

Assign discountPercent = 0.06

Else totalSpend < 100.00

Assign discountPercent = 0

Input total

if (total > account_bal)

Display "Error! You do not have enough available funds to purchase items: "

else if(total <= account_bal)

Display "Thank you for your purchase! Your updated account information is below: "

Display "Customer name- customer: "

Display "Username- user_name: "

Display "Password- pass_word: "

Display "Account number- account_num: "

Display "Member level- memberlvl: "

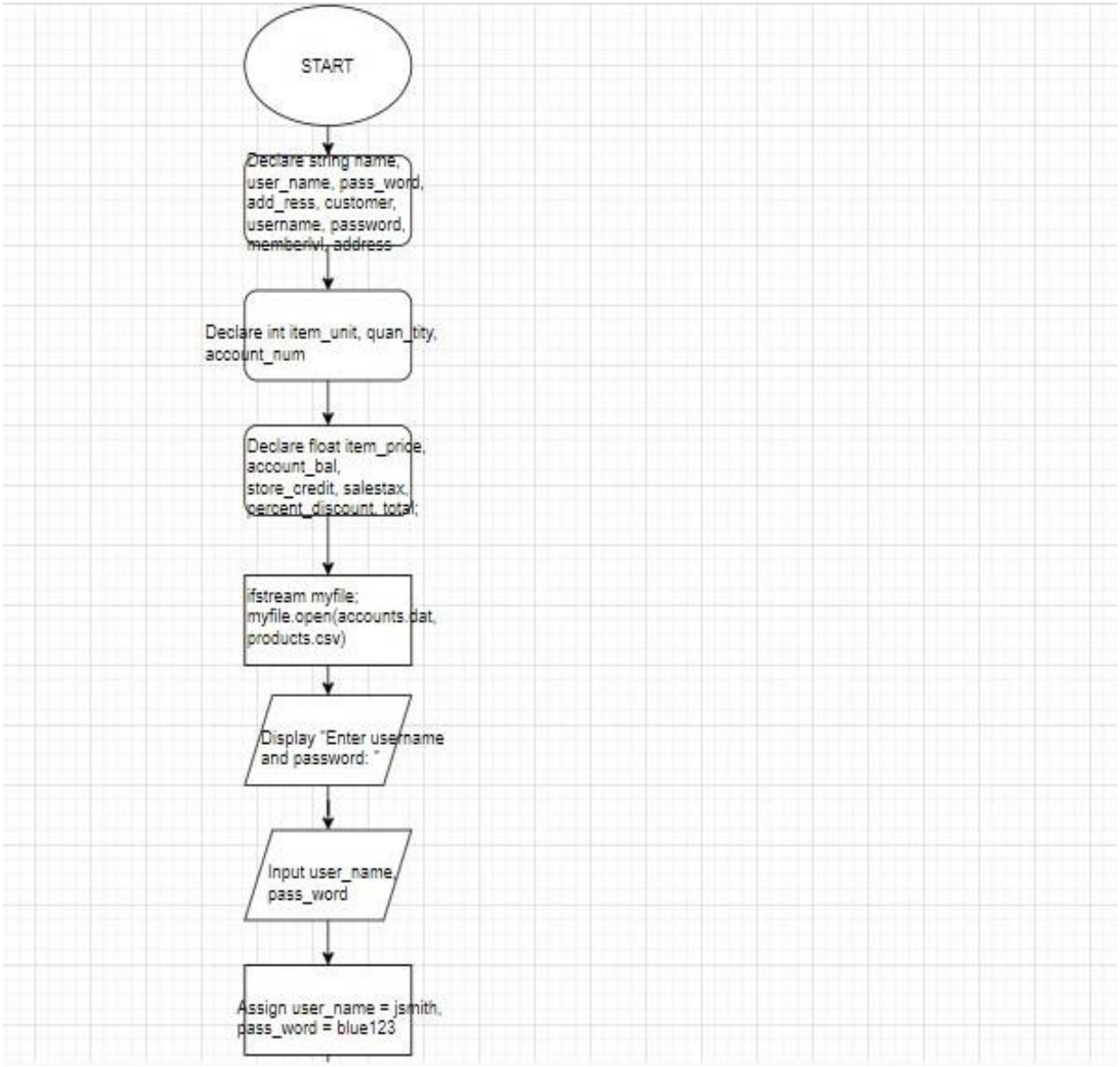
Display "Store Credit- NewStoreCredit: "

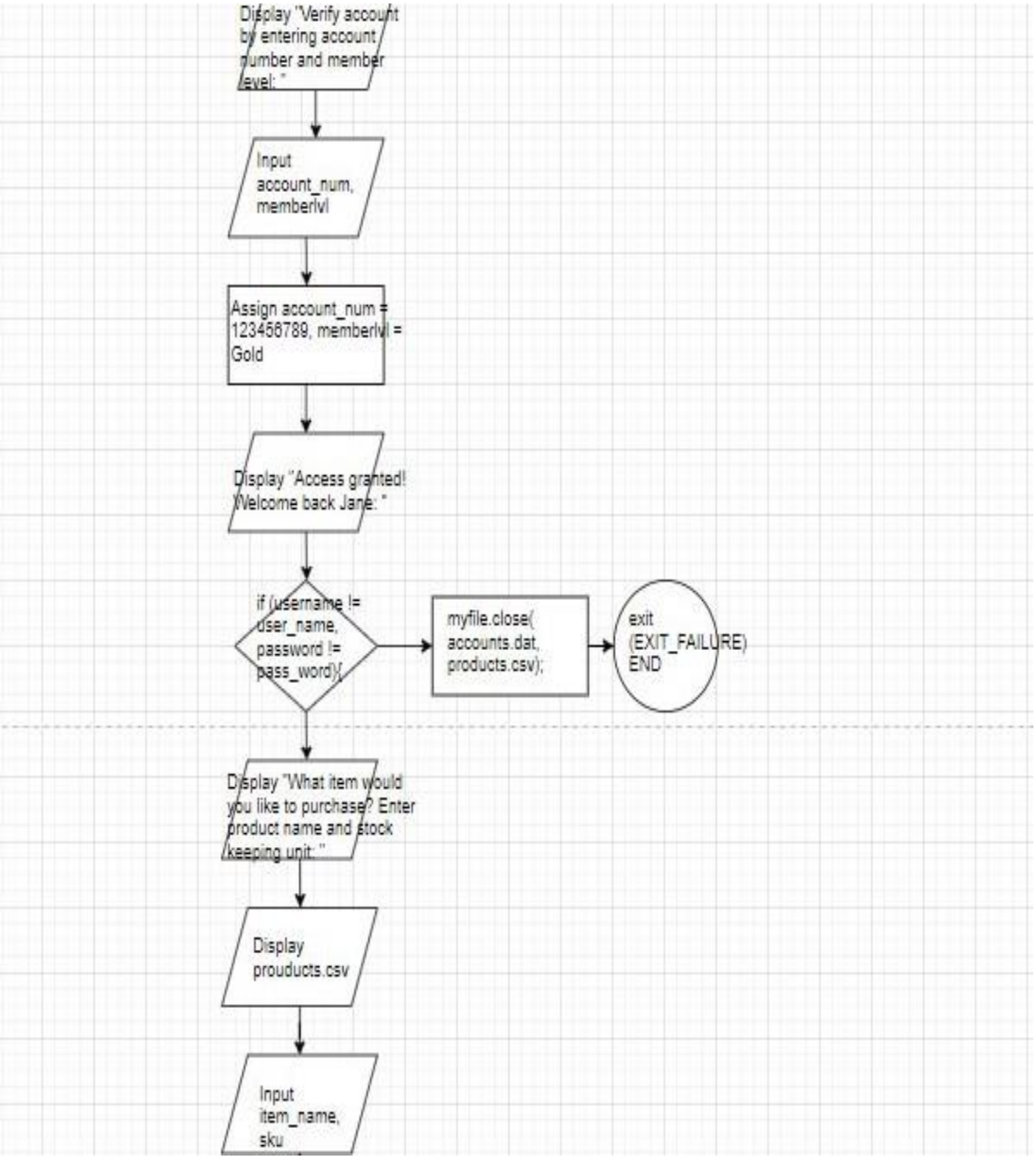
Assign NewStoreCredit = account_bal - total

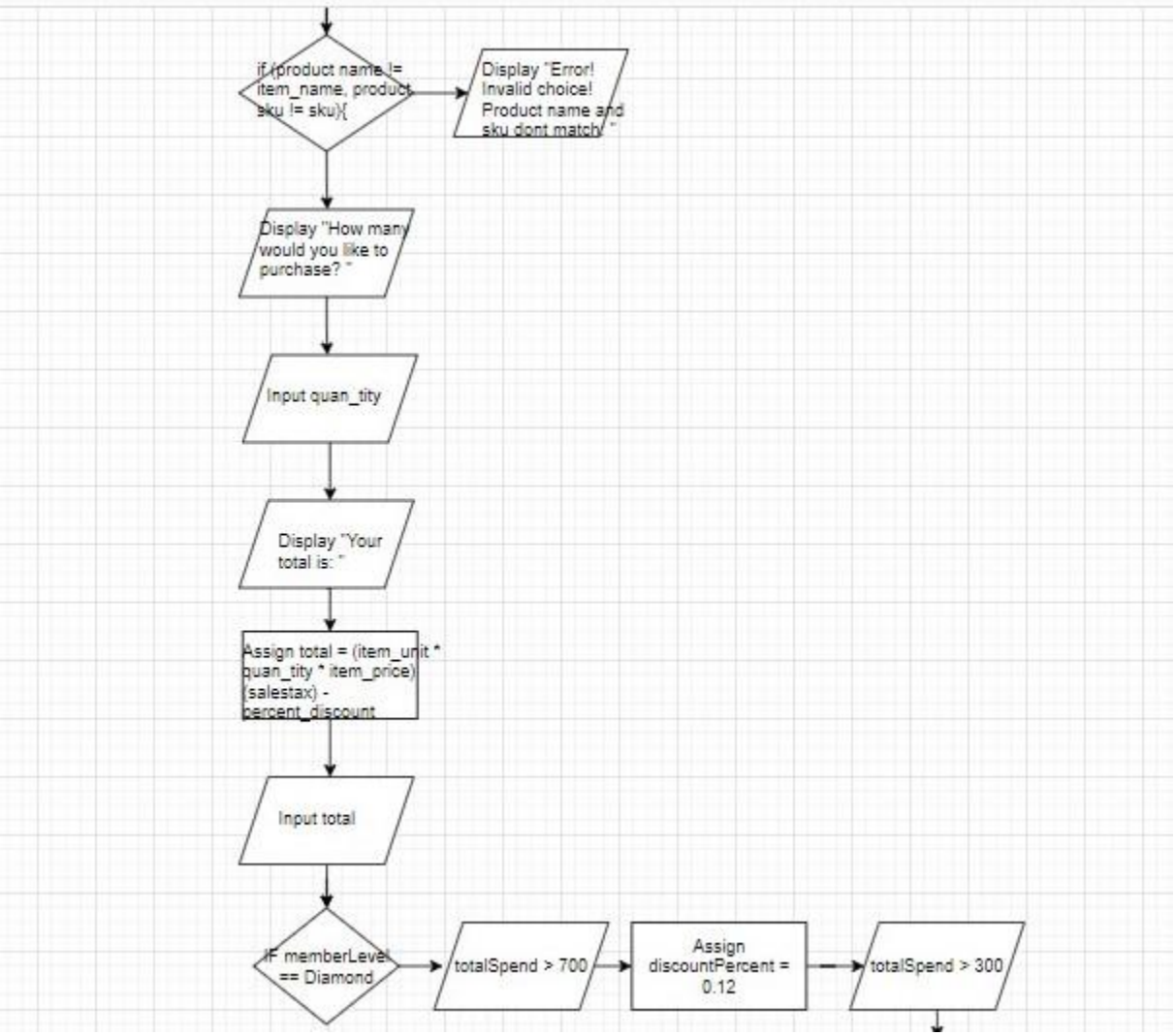
exit (EXIT_SUCCESS)

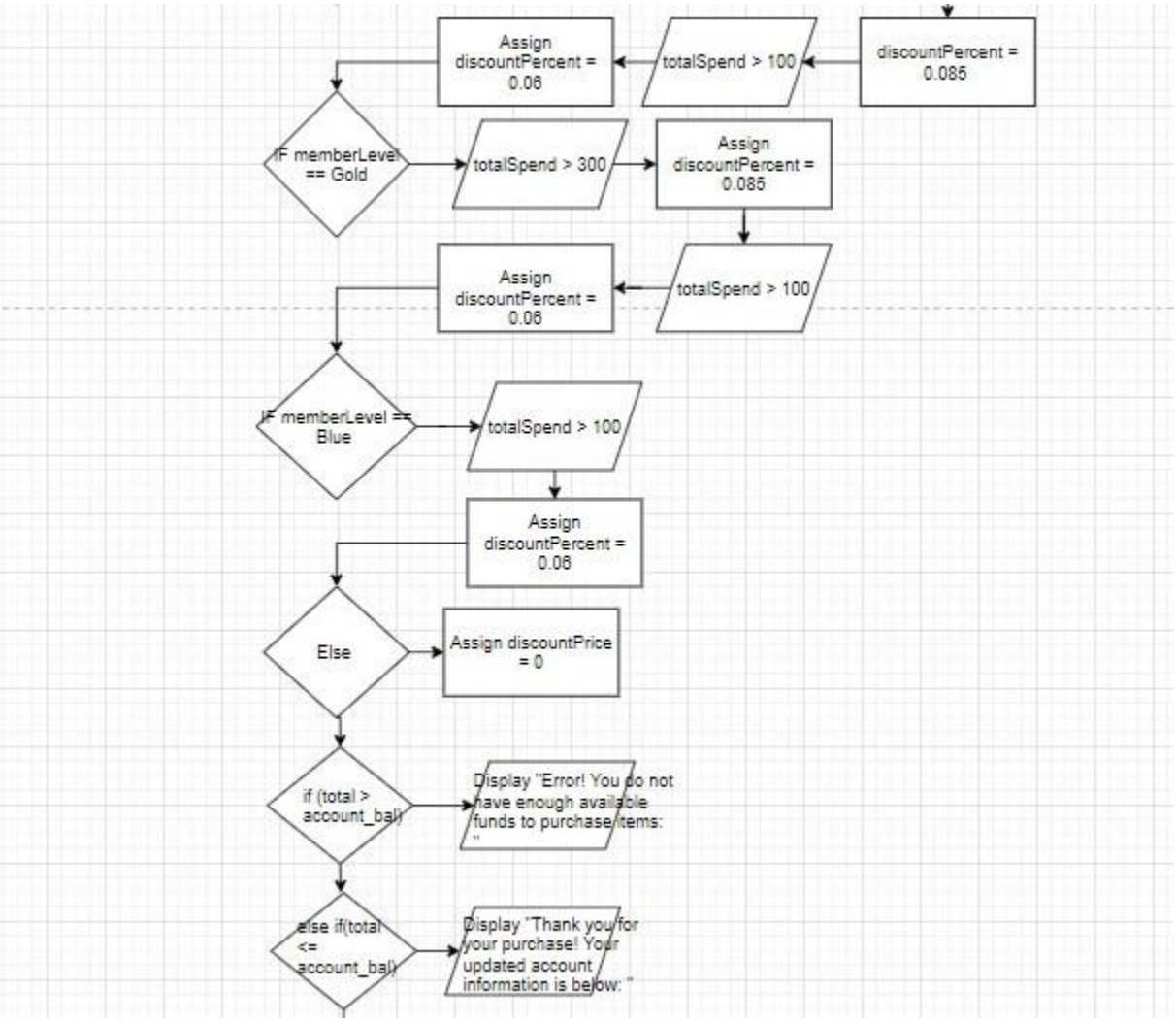
END

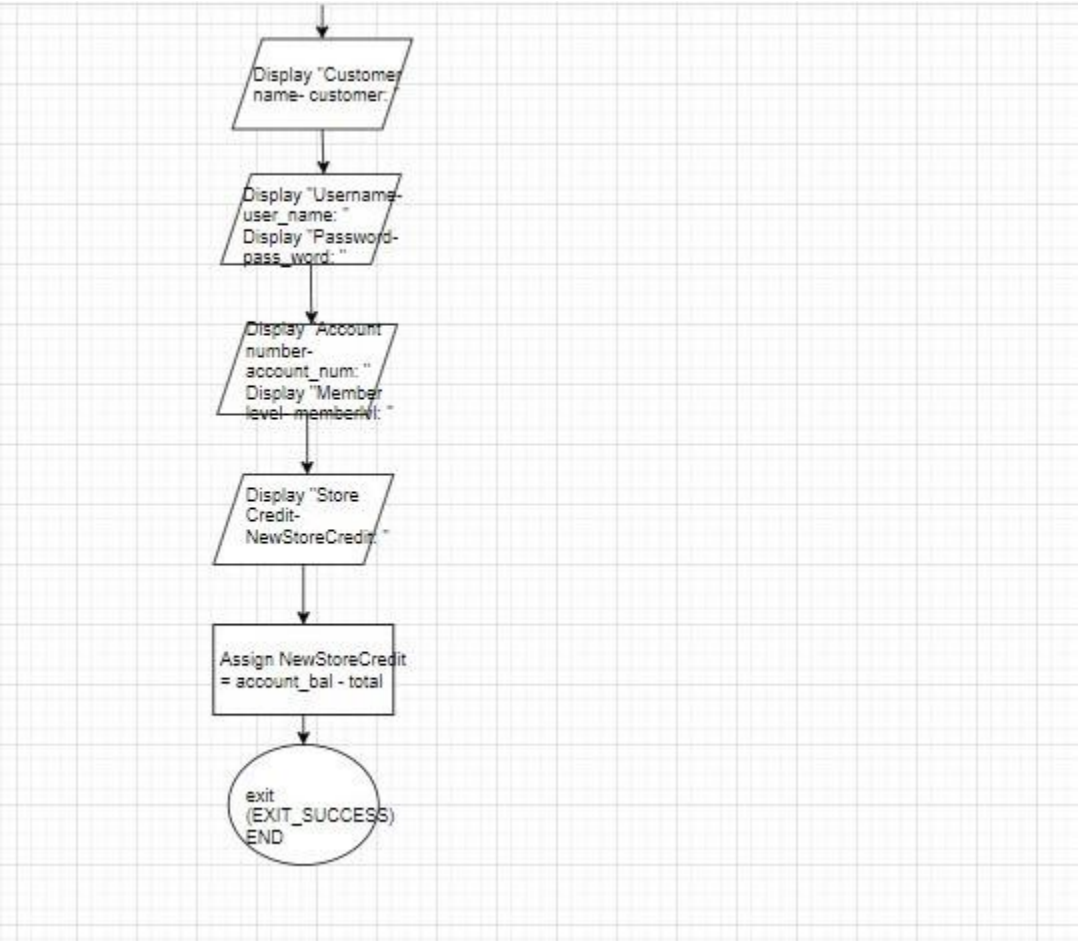
Flowchart


START
Declare string name, user_name, pass_word, add_ress, customer, username, password, memberivl, address
Declare int item_unit, quan_tity, account_num
Declare float item_pride, account_bal, store_credit, salestax, percent_discount, total;
ifstream myfile; myfile.open(accounts.dat, products.csv)
Display "Enter username and password: "
Input user_name, pass_word
Assign user_name = jsmith, pass_word = blue123


Display "Verify account by entering account number and member level: "
Input account_num, memberlvl
Assign account_num = 123456789, memberlvl = Gold
Display "Access granted! Welcome back Jane: "
if (username != user_name, password != pass_word){
myfile.close(accounts.dat, products.csv);
exit (EXIT_FAILURE) END
Display "What item would you like to purchase? Enter product name and stock keeping unit: "
Display prouducts.csv
Input item_name, sku


if (product name != item_name, product sku != sku){
Display "Error! Invalid choice! Product name and sku dont match! "
Display "How many would you like to purchase? "
Input quan_tity
Display "Your total is: "
Assign total = (item_unit * quan_tity * item_price) (salestax) - percent_discount
Input total
IF memberLevel == Diamond
totalSpend > 700
Assign discountPercent = 0.12
totalSpend > 300


discountPercent = 0.085
totalSpend > 100
Assign discountPercent = 0.06
IF memberLevel == Gold
totalSpend > 300
Assign discountPercent = 0.085
totalSpend > 100
Assign discountPercent = 0.06
IF memberLevel == Blue
totalSpend > 100
Assign discountPercent = 0.06
Else
Assign discountPrice = 0
if (total > account_bal)
Display "Error! You do not have enough available funds to purchase items: "
else if(total <= account_bal)
Display "Thank you for your purchase! Your updated account information is below: "


Display "Customer name- customer: "
Display "Username- user_name: "
Display "Password- pass_word: "
Display "Account number- account_num: "
Display "Member level- memberlvl: "
Display "Store Credit- NewStoreCredit: "
Assign NewStoreCredit = account_bal - total
exit (EXIT_SUCCESS) END

Test Cases

SKU	Name	Items in Unit	Price Per Unit	Quantity Purchased	Total	Result	Account Type	Percent Discount	Member Level Total	Store Credit
HF-342	$\frac{1}{2}$ in bolt	50	20.00	1	$1060.00	Pass	Gold	8.5	$969.90	$2030.10
LK-322	$\frac{1}{4}$ in nail	25	5.75	3	$457.13	Pass	Gold	8.5	$418.27	$2581.73
KF-231	Hammer	1	15.23	11	$177.58	Pass	Gold	6	$166.93	$2833.07
HF-342	Hammer	1	20.00			Fail. SKU and product name don't match	Gold	N/A	N/A	3000.00
HF-342	$\frac{1}{2}$ in bolt	50	20.00	107		Fail. Only a 100 in stock	Gold	N/A	N/A	3000.00
KF-231	Hammer	1	15.23	100	$1614.38	Pass	Gold	8.5	1477.16	
LK-322	$\frac{1}{4}$ in nail	25	5.75	36	$5485.50	Fail. Inefficient funds to make purchase.	Gold	N/A	N/A	

HF-342	$\frac{1}{2}$ in bolt	50	20.00	3	$3180	Pass	Gold	8.5	$2909.70	$90.30
KF-231	Hammer	1	15.23	15	$242.16	Pass	Gold	6	$227.63	$2772.37
LK-322	$\frac{1}{4}$ in nail	25	5.75	1	$152.38	Pass	Gold	6	$143.24	$2856.76

Code

```
#include <iostream>
#include <iomanip>
#include <string>
#include <fstream>
#include <stdlib.h>

int main()
{
   // declaring variables
   string name, user_name, pass_word, add_ress, customer, username, password,
memberlvl, address;
   int item_unit, quan_tity, account_num;
   float item_price, account_bal, store_credit, salestax, discountPercent,
totalSpend, total;
totalSpend = (item_price * item_unit * item_price)

 // open files contating account numbers and product information

   ifstream myfile;

   myfile.open(accounts.dat, products.csv);
cout << "Enter username and password: " << endl; // asks user to enter username
and password
cin >> user_name, pass_word; // user enter username and password
user_name = jsmith, pass_word = blue123; // assigns username and password
cout << "Verify account by entering account number and member level: " << endl;
// asks user to verify account
```

```
cin >> account_num, memberlvl; // user enters account number and member level to verify account
account_num = 123456789, memberlvl = Gold; // assigns account number and member level
cout << "Access granted! Welcome back Jane: " << endl; // user successfully logs into account
// if the username and password entered does not match it will fail and cause program to terminate
if (username != user_name, password != pass_word){
   myfile.close( accounts.dat, products.csv);
   EXIT_FAILURE;
}
cout<< "What item would you like to purchase? Enter product name and stock keeping unit: " << endl; // asks user what item they would like to purchase
cout << prouducts.csv << endl; // displays all items available for purchase
cin >> item_name, sku; // user enters item name and sku of item he would like to purchase
// if product name and product sku do not match this will yield an error message
if (product name != item_name, product sku != sku){
   cout << "Error! Invalid choice! Product name and sku dont match: " << endl;
}
cout << "How many would you like to purchase? " << endl;
cin >> quan_tity;
cout << "Your total is: " << endl;
total = (item_unit * quan_tity * item_price)(salestax) - percent_discount;
 if( memberLevel == "Diamond" ) {
 if( totalSpend > 700.0 ) {
 discountPercent = 0.12; // if user spends at least $700 they will get 12% discount
 } else if( totalSpend > 300.0 ) {
 discountPercent = 0.085; // if user spends at least $300 they will get 8.25% discount
 } else if( totalSpend > 100.0 ) {
 discountPercent = 0.06; // if user spends at least $100 they will get 6% discount
 }
 } else if( memberLevel == "Gold" ) {
 if( totalSpend > 300.0 ) {
```

```
 discountPercent = 0.085; // if user spends at least $300 they will get 8.25%
discount
 } else if( totalSpend > 100.0 ) {
 discountPercent = 0.06; // if user spends at least $100 they will get 6% discount
 }
 } else if( memberLevel == "Blue" ) {
 if( totalSpend > 100.0 ) {
 discountPercent = 0.06; // if user spends at least $100 they will get 6% discount
 }
 } else { // if user doesn't spend any of the above amounts then they get no
discount
 discountPercent = 0; // assign discount price to zero
cin >> total;
if (total > account_bal){
   cout << "Error! You do not have enough available funds to purchase items: " <<
endl;
   else if(total <= account_bal){
      cout << "Thank you for your purchase! Your updated account information is
below: " << endl; }
}
// displays updated account information after purchase
cout << "Customer name- customer: " << endl;
cout << "Username- user_name: " << endl;
cout << "Password- pass_word: " << endl;
cout << "Account number- account_num: " << endl;
cout << "Member level- memberlvl: " << endl;
cout << "Store Credit- NewStoreCredit: " << endl;
NewStoreCredit = account_bal - total;
exit (EXIT_SUCCESS); // successfully logs user out after showing updated
information
}
return 0;
```

Grade Sheet

Fundamentals of Programming
Ms. Vanessa Coote

Before submitting the project package, the student should review each of the elements listed below and put a checkmark only in those checkboxes where the designated elements has been reviewed and meets specifications. After completing your document package, number your pages and write the designated page numbers onto the spaces provided on the grading sheet.

_____ Professionalism (10 points)

- ☐ Following directions
- ☐ Neatly assembled 8 ½ by 11
- ☐ Cover page
- ☐ Page numbers
- ☐ Documentation

_____ Source Program Listing and Proper Execution of Program (30 points)
It is expected that each student's program will run correctly

- ☐ Program source code listing matches code on submission and/or backups
- ☐ Inclusion of comment lines in source code
  - ~ Comments at the beginning of the program including programmer, project name and number, date written, and brief program description.
  - ~ Comments at key locations throughout the code
- ☐ Descriptive variable names (that follow naming convention)
- ☐ Logic is correct
- ☐ Logic is clear and easy to follow
- ☐ Proper formatting of statements
- ☐ Alignment, proper indentation, etc
- ☐ Proper use of data types and data conversions

_____ Test Data (5 points)

- ☐ Each test case properly calculated by hand and documented
- ☐ Suitable choice of you own test data case

_____ Input Window (10 points)

- ☐ Correct data type for each input section
- ☐ Analysis of data type (e.g. int, float, double etc.)
- ☐ Appropriate restrictions for each input section
- ☐ Data input value shown matches specified test data
- ☐ Appropriate display for each input section

_____ Output (15 points)

- ☐ Suitable layout of output (including required fields, easy to read layout, etc.)
- ☐ All data cases displayed
- ☐ Correct value displayed for each case
- ☐ Correct format of fields (e.g. use of integers and not float as appropriate, dollars and cents, display of $, etc)
- ☐ Required output format
- ☐ Aesthetics (User-friendliness, easy to understand output, alignments, etc)

_____ Documentation (40 points)

- ☐ Analysis of specifications
- ☐ Pseudocode
- ☐ Flowchart
- ☐ Hard copy of program

_____ Fully Functioning Program (30 points)

_____

Possible points = 140

Points Earned =