Assignment 5 STL using C++

Krishnaraj, pa20, sycsf

// Shop has item code, item name, quantity, and the cost.

// Input would be some item name, where we would first add things to the shop inventory

// Another option would be to check for the item in the database which is a list.

// If the item is found in the shop then you are supposed to ask them the number of items.

// if its available then you sell it, or else you tell them that that may items arent in stock.

#include <iostream>

#include <list>

using namespace std;

// struct so we can put it in a single linked list.

struct Items

{

int item\_code;

string item\_name;

int item\_quantity;

int item\_cost;

Items(int a, string b, int q, int c) : item\_code(a), item\_name(b), item\_quantity(q), item\_cost(c)

{

}

} currentItem(0, "", 0, 0), itemToAdd(0, "", 0, 0);

// linked list items so we can put objects of structures in it easily.

list<Items> items = {

Items(101, "Burger", 10, 150),

Items(102, "Fries", 10, 129),

Items(103, "Ice Cream", 10, 40),

Items(104, "Coke", 10, 50),

};

bool itemFound = false;

// inserts items in the list

void insertItems()

{

cout << "Enter the details of the Item that you wanna enter to the database" << endl;

cout << "Enter the Code of the new Item: ";

cin >> itemToAdd.item\_code;

cout << "Enter the Name of the new Item: ";

cin >> itemToAdd.item\_name;

cout << "Enter the Quantity of the new Item: ";

cin >> itemToAdd.item\_quantity;

cout << "Enter the Cost of the new Item: ";

cin >> itemToAdd.item\_cost;

items.push\_back(itemToAdd);

cout << "Item added successfully" << endl;

}

// returns true or false depending on whether the item was found

bool searchItem(int itemCode)

{

int i = 0;

for (list<Items>::iterator it = items.begin(); it != items.end(); it++, i++)

{

struct Items temp = \*it;

if (temp.item\_code == itemCode)

{

currentItem = temp;

return true;

}

}

return false;

}

// just find the element at the element currentItemIndex, and replace it with the currentItem struct object.

void updateItems(int quantity, int currentItemCode)

{

for (list<Items>::iterator it = items.begin(); it != items.end(); it++)

{

struct Items temp = \*it;

if (temp.item\_code == currentItemCode)

{

currentItem.item\_quantity -= quantity;

items.erase(it);

items.push\_back(currentItem);

}

}

}

// display the things, and check if the selected item code by the user exists in the thing by calling searchItem. Then input the item quantity, check for it, and update the currentItem object.

// then call the updateItem function.

bool displayAndPurchaseItems()

{

int selectedItemCode = -1;

int selectedQuantity = 0;

struct Items tempItem(0, "", 0, 0);

cout << "The Items that you can buy are: " << endl;

for (list<Items>::iterator it = items.begin(); it != items.end(); it++)

{

tempItem = \*it;

cout << tempItem.item\_code << " " << tempItem.item\_name << " " << tempItem.item\_quantity << " " << tempItem.item\_cost << endl;

}

c:

try

{

cout << "Please enter the code of the item that you wanna buy" << endl;

cin >> selectedItemCode;

if (!searchItem(selectedItemCode))

{

throw selectedItemCode;

}

l:

cout << "Enter the Quantity of the Item that you wanna buy. "

<< endl;

cout << "Max quantity is: " << currentItem.item\_quantity << endl;

cin >> selectedQuantity;

try

{

if (selectedQuantity > currentItem.item\_quantity)

{

throw selectedQuantity;

}

else

{

cout << "Thank you for Purchasing!" << endl;

updateItems(selectedQuantity, selectedItemCode);

}

}

catch (int something)

{

cout << "Quantity you entered is too much! Try again!" << endl;

goto l;

}

return true;

}

catch (int something)

{

cout << "The code of the item that you entered doesnt exist. Try again. " << endl;

goto c;

}

return true;

}

int main()

{

int choice = 0;

struct Items;

cout << "Welcome to McRonalds" << endl;

do

{

cout << endl

<< "What do you wanna do?\n\

1. Add new Items\n\

2. Purchase Item\n\

3. Quit\n"

<< endl;

cin >> choice;

switch (choice)

{

case 1:

insertItems();

break;

case 2:

if (!displayAndPurchaseItems())

{

cout << "Item couldnt be purchased!" << endl;

}

else

{

cout << "Item purchased and Updated Successfully" << endl;

}

break;

case 3:

cout << "Thanks for Visiting our store!" << endl;

break;

default:

cout << "Sorry, we cant do that in this store" << endl;

break;

}

itemFound = false;

} while (choice != 3);

return 0;

}