Departmental Store Management System

Group 10
Nicolas Wiesner, Wayne Seide

Programming 1 COP 2006
Deepa Devasenapathy

Goals of the Project and Software requirements

Goals

What will the program be able to do:

- Enter products and display that data to the user
- Editing products either at an existing location
- Displaying the sales at the end of the day

Software Requirements:

- **Operating System:**
- Compatible with Windows, macOS, or Linux.

Memory (RAM):

- Minimum: 512 MB.
- Recommended: 2 GB or more for smoother operation with IDEs.

Storage:

- A few MBs of free space for compiler and IDE installation or Online GBD

Why did we choose this project?

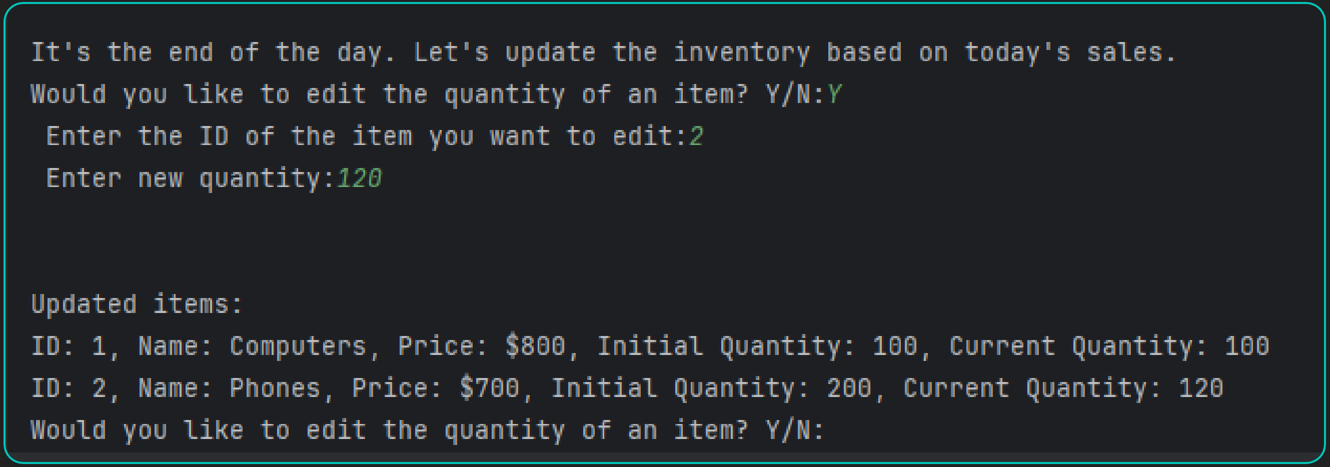
We wanted to learn how to create a real-world application that people or businesses could use to accomplish something.

Being able to manage updates and information is an important part that most companies are in need of, from accounting to store management, like in this project.

Being able to apply what we learned in class to a real-world problem was an important part in us choosing this application.

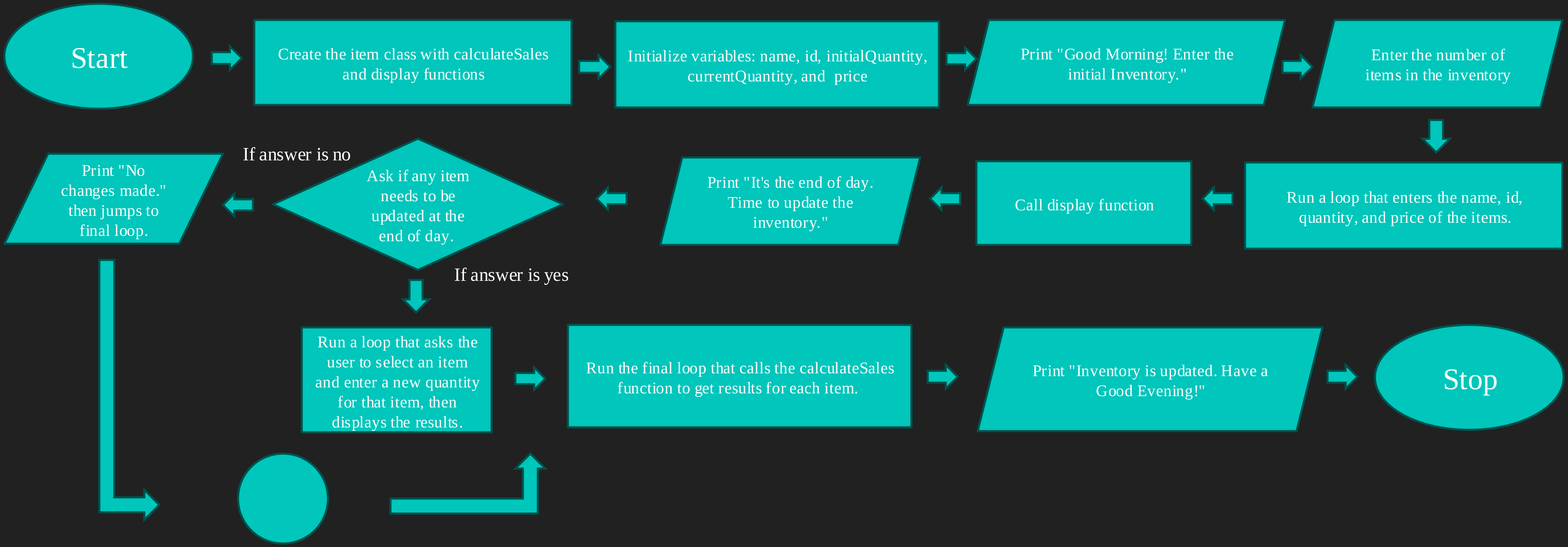
Existing Techniques and Improvements

- In order to manage data, we developed a project that uses vectors. We also made it possible to update the details about specific objects and adjust their quantity so that we can observe changes from the start of the day to the end of the day.



```
It's the end of the day. Let's update the inventory based on today's sales.
Would you like to edit the quantity of an item? Y/N:Y
 Enter the ID of the item you want to edit:2
 Enter new quantity:120


Updated items:
ID: 1, Name: Computers, Price: $800, Initial Quantity: 100, Current Quantity: 100
ID: 2, Name: Phones, Price: $700, Initial Quantity: 200, Current Quantity: 120
Would you like to edit the quantity of an item? Y/N:
```

Flow Diagram

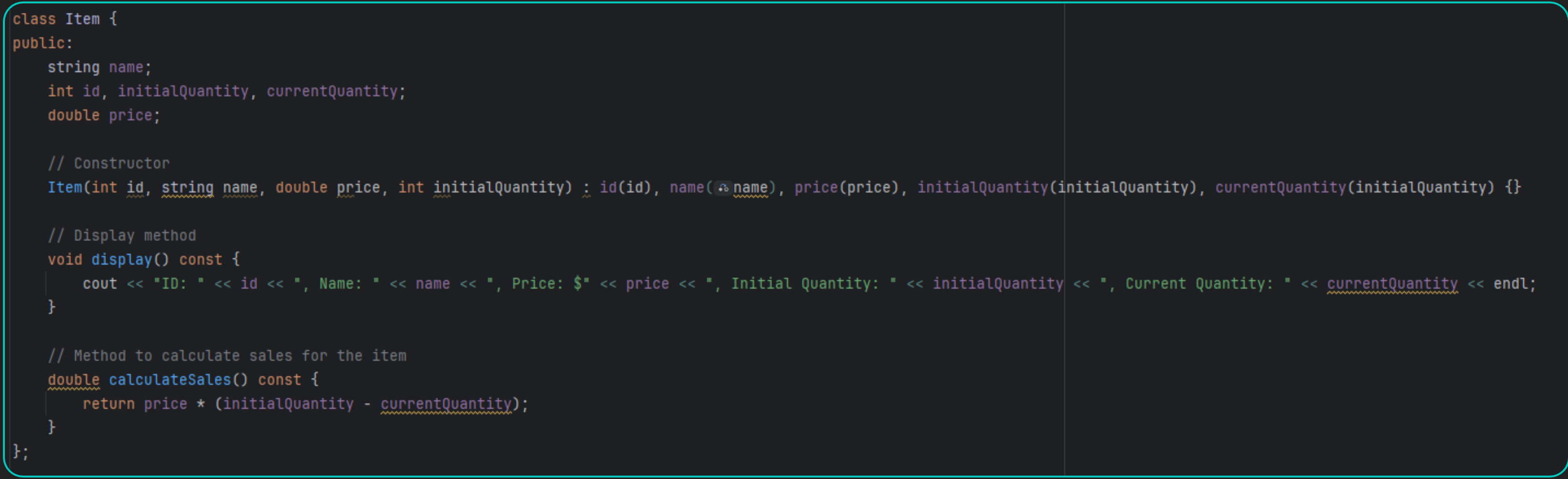

Start
Create the item class with calculateSales and display functions
Initialize variables: name, id, initialQuantity, currentQuantity, and price
Print "Good Morning! Enter the initial Inventory."
Enter the number of items in the inventory
Run a loop that enters the name, id, quantity, and price of the items.
Call display function
Print "It's the end of day. Time to update the inventory."
Ask if any item needs to be updated at the end of day.
If answer is no
Print "No changes made." then jumps to final loop.
If answer is yes
Run a loop that asks the user to select an item and enter a new quantity for that item, then displays the results.
Run the final loop that calls the calculateSales function to get results for each item.
Print "Inventory is updated. Have a Good Evening!"
Stop

How our project works:

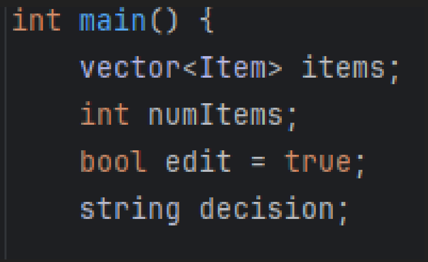
To store data, we used a vector that holds a class of each item's information, including:

- ID
- Name
- Price
- Initial Quantity
- Current quantity

We use initial quantity and current quantity to display the changes made at end of the day.



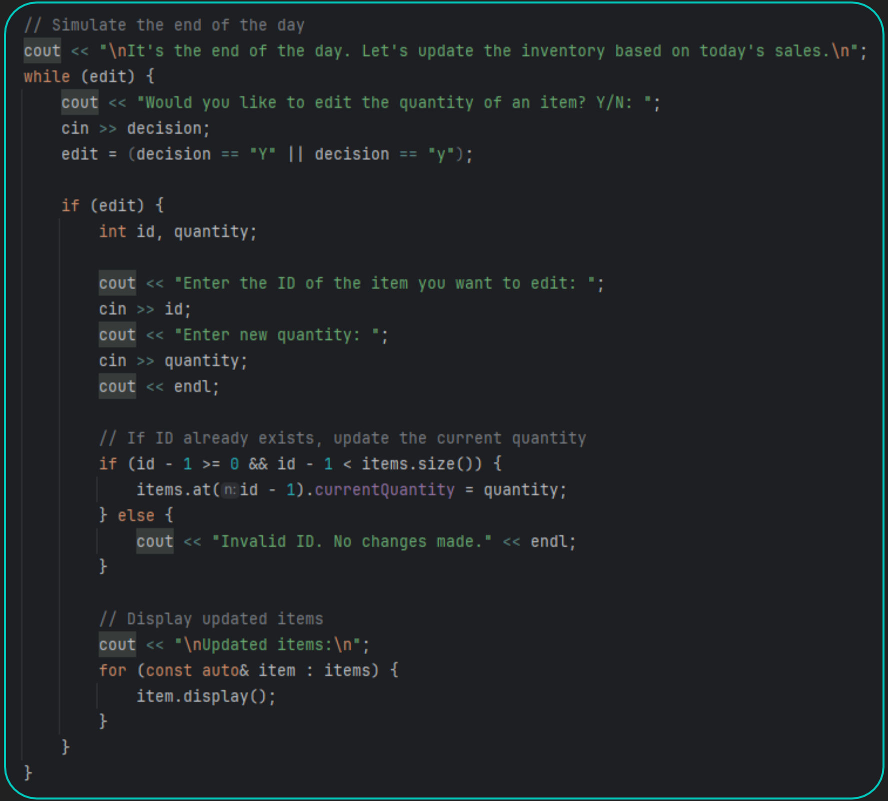
```cpp
class Item {
public:
    string name;
    int id, initialQuantity, currentQuantity;
    double price;

    // Constructor
    Item(int id, string name, double price, int initialQuantity) : id(id), name( name), price(price), initialQuantity(initialQuantity), currentQuantity(initialQuantity) {}

    // Display method
    void display() const {
        cout << "ID: " << id << ", Name: " << name << ", Price: $" << price << ", Initial Quantity: " << initialQuantity << ", Current Quantity: " << currentQuantity << endl;
    }

    // Method to calculate sales for the item
    double calculateSales() const {
        return price * (initialQuantity - currentQuantity);
    }
};
```



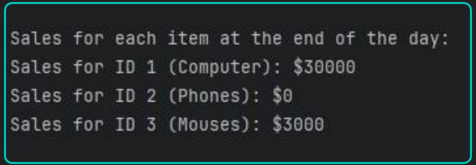
```cpp
int main() {
    vector<Item> items;
    int numItems;
    bool edit = true;
    string decision;
```

How our project works:

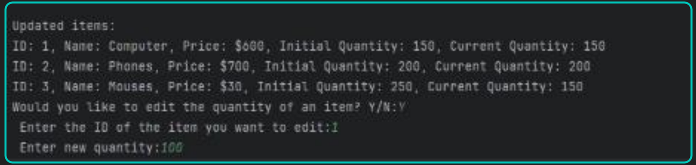
- Using the stored information, we can display the changes as well as the sales of each product at the end of the day



```cpp
// Simulate the end of the day
cout << "\nIt's the end of the day. Let's update the inventory based on today's sales.\n";
while (edit) {
    cout << "Would you like to edit the quantity of an item? Y/N: ";
    cin >> decision;
    edit = (decision == "Y" || decision == "y");

    if (edit) {
        int id, quantity;

        cout << "Enter the ID of the item you want to edit: ";
        cin >> id;
        cout << "Enter new quantity: ";
        cin >> quantity;
        cout << endl;

        // If ID already exists, update the current quantity
        if (id - 1 >= 0 && id - 1 < items.size()) {
            items.at( n: id - 1).currentQuantity = quantity;
        } else {
            cout << "Invalid ID. No changes made." << endl;
        }

        // Display updated items
        cout << "\nUpdated items:\n";
        for (const auto& item : items) {
            item.display();
        }
    }
}
```

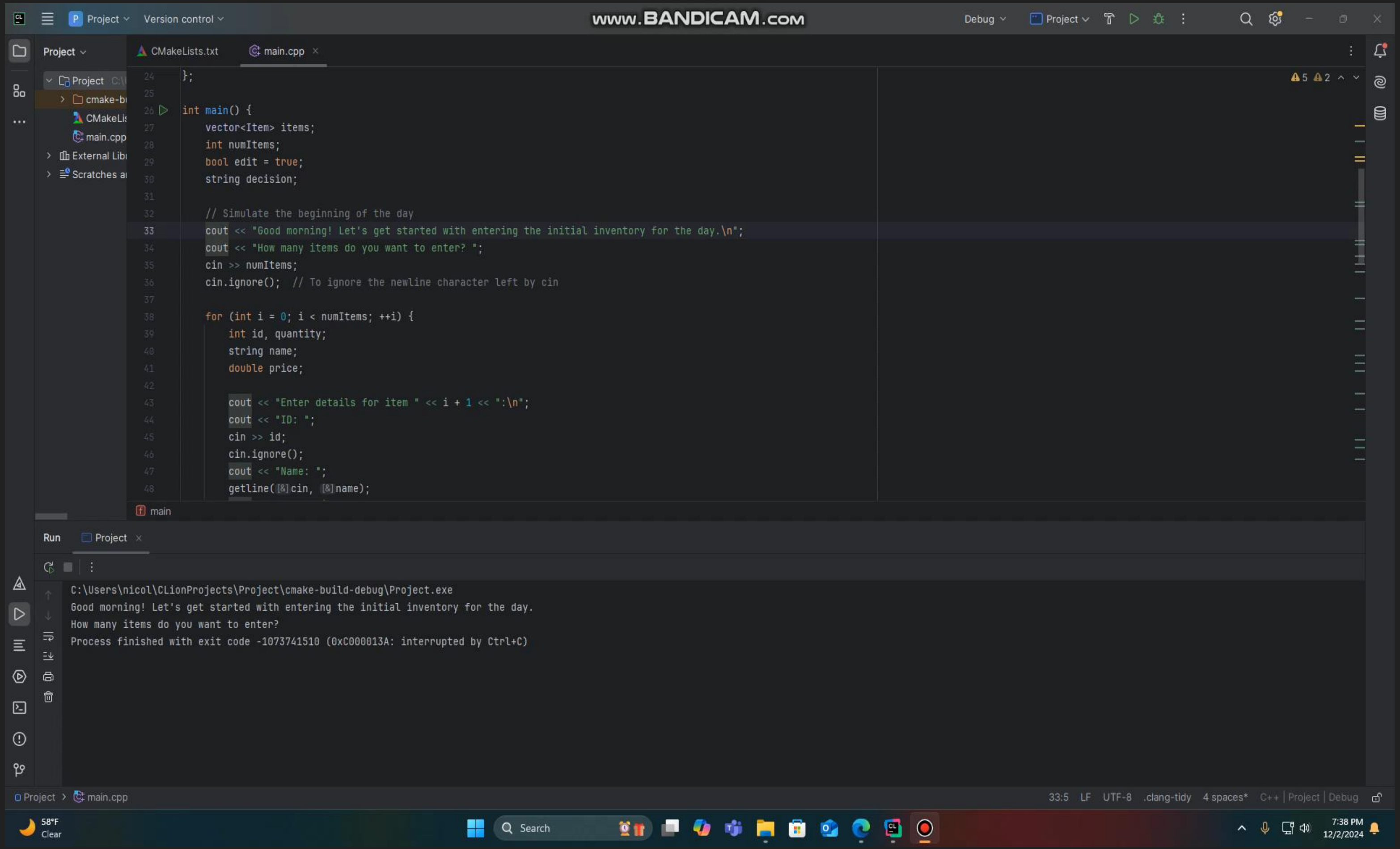


```
Sales for each item at the end of the day:
Sales for ID 1 (Computer): $30000
Sales for ID 2 (Phones): $0
Sales for ID 3 (Mouses): $3000
```



```
Updated items:
ID: 1, Name: Computer, Price: $600, Initial Quantity: 150, Current Quantity: 150
ID: 2, Name: Phones, Price: $700, Initial Quantity: 200, Current Quantity: 200
ID: 3, Name: Mouses, Price: $30, Initial Quantity: 250, Current Quantity: 150
Would you like to edit the quantity of an item? Y/N:Y
 Enter the ID of the item you want to edit:1
 Enter new quantity:100
```

Video of the code working



```cpp
};

int main() {
    vector<Item> items;
    int numItems;
    bool edit = true;
    string decision;

    // Simulate the beginning of the day
    cout << "Good morning! Let's get started with entering the initial inventory for the day.\n";
    cout << "How many items do you want to enter? ";
    cin >> numItems;
    cin.ignore();  // To ignore the newline character left by cin

    for (int i = 0; i < numItems; ++i) {
        int id, quantity;
        string name;
        double price;

        cout << "Enter details for item " << i + 1 << ":\n";
        cout << "ID: ";
        cin >> id;
        cin.ignore();
        cout << "Name: ";
        getline(cin, name);
```

```
C:\Users\nicol\CLionProjects\Project\cmake-build-debug\Project.exe
Good morning! Let's get started with entering the initial inventory for the day.
How many items do you want to enter?
Process finished with exit code -1073741510 (0xC000013A: interrupted by Ctrl+C)
```

Conclusion:

The project is a good example of what we could accomplish but I feel like there are some improvements we could make to simulate a department store more:

- Being able to save the data In a file
- Being able to add more inventory
- Changing the price of an item
- Getting rid of a items
- Exception handling

Any Questions?