

CST-250 Activity 1

Kimberly Alvarez

3/19/2025

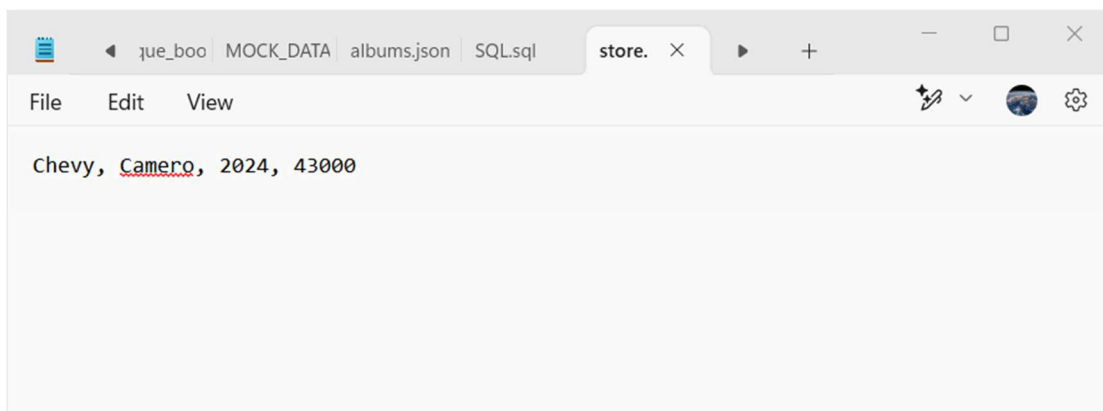
Part 2 – Console Application

```
C:\Users\kima\source\repos\ x + v
Welcome to the car shop! First you must create some cars and put them into the inventory. Then you may add cars to the cart. Finally, you may checkout, which will calculate your total bill.
Choose an action: (0) quit (1) create car (2) add car to cart (3) checkout (4) save inventory to a text file (5) Load inventory from text file.
1
Enter the make of the car:
Ford
Enter the model of the car:
Mustang
Enter the year of the car:
2025
Enter the price of the car:
73000
Inventory:
0: 2025 Ford Mustang - $73000
Choose an action: (0) quit (1) create car (2) add car to cart (3) checkout (4) save inventory to a text file (5) Load inventory from text file.
1
Enter the make of the car:
Toyota
Enter the model of the car:
Camary
Enter the year of the car:
2025
Enter the price of the car:
45000
Inventory:
0: 2025 Ford Mustang - $73000
1: 2025 Toyota Camary - $45000
Choose an action: (0) quit (1) create car (2) add car to cart (3) checkout (4) save inventory to a text file (5) Load inventory from text file.
2
Inventory:
0: 2025 Ford Mustang - $73000
1: 2025 Toyota Camary - $45000
Enter the index of the car you would like to add to the cart:
0
Inventory:
0: 2025 Ford Mustang - $73000
1: 2025 Toyota Camary - $45000
Choose an action: (0) quit (1) create car (2) add car to cart (3) checkout (4) save inventory to a text file (5) Load inventory from text file.
2
Inventory:
0: 2025 Ford Mustang - $73000
1: 2025 Toyota Camary - $45000
Enter the index of the car you would like to add to the cart:
1
Inventory:
0: 2025 Ford Mustang - $73000
1: 2025 Toyota Camary - $45000
Choose an action: (0) quit (1) create car (2) add car to cart (3) checkout (4) save inventory to a text file (5) Load inventory from text file.
3
Shopping List:
0: 2025 Ford Mustang - $73000
1: 2025 Toyota Camary - $45000
Your total is: 118000
Inventory:
0: 2025 Ford Mustang - $73000
1: 2025 Toyota Camary - $45000
Choose an action: (0) quit (1) create car (2) add car to cart (3) checkout (4) save inventory to a text file (5) Load inventory from text file.
```

In the image above we have created a console application. The application begins by prompting the user to decide what action they wish to be in the program. In this image we can see that we began by choosing action 1 which is the creation of a car. After building the car the inventory prints out the car information in the system so far. In the example set above we created two cars. The next thing that we did was add these cars to our cart by typing action number 2 and adding each car by their index number provided in the inventory list. Then we are able to see what the total cost for both vehicles would be. While completing this exercise I did have some trouble as I kept getting `CarLibrary.Car` in my inventory rather than the cars' information to the cars I was building. I had to do some research on how to fix this issue and I used Microsoft Copilot to help me determine what could be causing the issue. It turned out that I needed to add an override string in my `Car` class to ensure that my data was readable.

Data Storage

```
C:\Users\kima\source\repos\ x + v
Welcome to the car shop! First you must create some cars and put them into the inventory. Then you may add cars to the c
art. Finally, you may checkout, which will calculate your total bill.
Choose an action: (0) quit (1) create car (2) add car to cart (3) checkout (4) save inventory to a text file (5) Load in
ventory from text file.
1
Enter the make of the car:
Chevy
Enter the model of the car:
Camero
Enter the year of the car:
2024
Enter the price of the car:
43000
Inventory:
0: 2024 Chevy Camero - $43000
Choose an action: (0) quit (1) create car (2) add car to cart (3) checkout (4) save inventory to a text file (5) Load in
ventory from text file.
4
Inventory:
0: 2024 Chevy Camero - $43000
Choose an action: (0) quit (1) create car (2) add car to cart (3) checkout (4) save inventory to a text file (5) Load in
ventory from text file.
```



In the first image we are creating a car and saving it to our inventory text file. In the second image we can see the inventory text file and the created car has been saved inside of it.

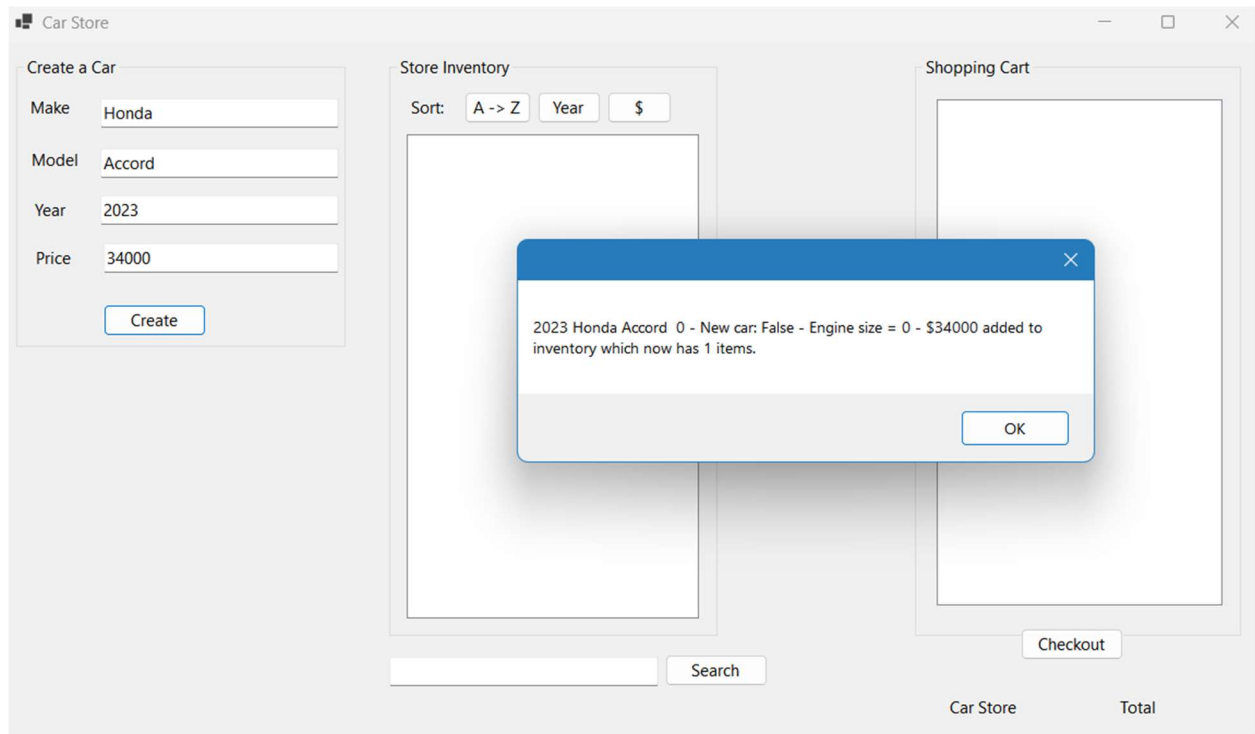
Console App Coding Challenge:

```
C:\Users\kima\source\repos\ x + v
Enter the model of the car:
Mustang
Enter the year of the car:
2024.5
Your input was invalid. Please enter a whole number as the year of the car.
Enter the year of the car:
2024
Enter the price of the car:
73000.999
Enter the color of the car:
Navy Blue
Enter the current milage of the car:
5000.5
Your input was invalid. Please enter mileage as a whole number.
Enter the current milage of the car:
5001
Is the car new? Enter Yes if new. No if used.
No
Enter the engine size:
6.5
Inventory:
0: 2024 Ford Mustang Navy Blue 5001 - New car: False - Engine size = 6.5 - $73000.999
Choose an action: (0) quit (1) create car (2) add car to cart (3) checkout (4) save inventory to a text file (5) Load inventory from text file.
4
Inventory:
0: 2024 Ford Mustang Navy Blue 5001 - New car: False - Engine size = 6.5 - $73000.999
Choose an action: (0) quit (1) create car (2) add car to cart (3) checkout (4) save inventory to a text file (5) Load inventory from text file.
```

```
store.txt
File Edit View
Ford, Mustang, 2024, 73000.999, Navy Blue, 5001, New:False, Engine size:6.5
```

In the image above we were asked to add try and catch to our actions just in case the user inputs an invalid value. The program could handle the issue without crashing or stopping the program's progress. As we can see in the image above, I input a decimal where only integers are valid values, this caused a prompt for the user to try inputting a new valid value. The program was able to proceed after a valid input was entered. We can also see that the values were all saved in the store.txt file of inventory. For this challenge I was able to go back to previous projects I have built previously and the provided example to model how to develop the try and catch while loops.

Part 3 – GUI



This image above demonstrates the application being implemented in a window format. The user is able to input the information for a car and once they press the create button. A secondary window opens and it lets the user know that the car has been added.

Add to Cart

The screenshot displays a web application titled "Car Store" with three main sections: "Create a Car", "Store Inventory", and "Shopping Cart".

- Create a Car:** Contains input fields for "Make", "Model", "Year", and "Price", along with a "Create" button.
- Store Inventory:** Features a "Sort" dropdown menu with options "A -> Z", "Year", and "\$". Below it is a list of cars, with the first item "2023 Honda Accord 0 - New car: False" highlighted in blue. A "Search" button is located at the bottom of this section.
- Add to Cart ->:** A button positioned between the "Store Inventory" and "Shopping Cart" sections.
- Shopping Cart:** Displays a list of items in the cart, with the first item "2023 Honda Accord 0 - New car: False" highlighted in blue. A "Checkout" button is located at the bottom of this section.

At the bottom right of the application, there are labels for "Car Store" and "Total".

In the image above we have added the created car into the store inventory after creation/input. Then the car is being added to cart through the use of the add to cart button. Finally, we can see that the car has been added to the shopping cart list.

Find Total Cost

The screenshot displays a web application titled "Car Store" with a light gray background. It is divided into three main sections: "Create a Car", "Store Inventory", and "Shopping Cart".

- Create a Car:** Located on the left, it contains four input fields labeled "Make", "Model", "Year", and "Price", each with a corresponding text box. Below these fields is a "Create" button.
- Store Inventory:** Located in the center, it features a "Sort:" dropdown menu with options "A -> Z", "Year", and "\$". Below the menu is a list of cars, with the first item "2023 Honda Accord 0 - New car: False" highlighted in blue. A "Search" button is positioned at the bottom of this section.
- Shopping Cart:** Located on the right, it contains a list of items in the cart, with the first item "2023 Honda Accord 0 - New car: False" highlighted in blue. Below the list is a "Checkout" button.

At the bottom right of the application, the text "Car Store" and "\$34,000.00" are displayed, indicating the total cost of the items in the cart.

In the image above we have activated the checkout button to display the total cost of the shopping cart. The total is displayed where the label with the text that had said total or label6 was displayed.

Part 4 – Search Button

The screenshot displays a web application titled "Car Store" with three main sections: "Create a Car", "Store Inventory", and "Shopping Cart".

- Create a Car:** Contains four input fields labeled "Make", "Model", "Year", and "Price", followed by a "Create" button.
- Store Inventory:** Features a "Sort:" dropdown menu with options "A -> Z", "Year", and "\$". Below it is a list of cars. The first item, "2025 toyota camry 0 - New car: False -", is highlighted in blue. At the bottom of this section is a search bar containing the text "toyota" and a "Search" button.
- Shopping Cart:** Contains a large empty rectangular area and a "Checkout" button at the bottom.

At the bottom right of the application, there are two labels: "Car Store" and "Total".

In the image above the user has been able to input cars and completed a search in the cars input to find a specific car. Originally I had inputted two cars, but after completing the search only one car remained. Now I did have some difficulty with this one as I was trying to follow an old assignment, but the issue was that in my previous assignment we searched a datagridview and it was throwing me off for this assignment. However, the auto generated fillers in VS helped where I was lost.

Sort By Make

The screenshot shows a web application titled "Car Store" with three main sections: "Create a Car", "Store Inventory", and "Shopping Cart".

- Create a Car:** Contains four input fields for "Make", "Model", "Year", and "Price", and a "Create" button.
- Store Inventory:** Features a "Sort:" dropdown menu with three options: "A -> Z" (selected), "Year", and "\$". Below the menu is a list of three cars:
 - 2023 honda accord 0 - New car: False -
 - 2022 tesla model x 0 - New car: False -
 - 2025 toyota camry 0 - New car: False -
- Shopping Cart:** An empty rectangular area with a "Checkout" button at the bottom.

At the bottom of the interface, there is a search bar with a "Search" button, and labels for "Car Store" and "Total".

In the image above I have sorted out the inventory list by make. I activated the sort by clicking the A -> Z sort button. This button can only put the names for the makes of the cars into alphabetical order. I was able to complete this by creating a Boolean that would check to see if the list was already in alphabetical order. I also had to add if and else statements to sort how the list needed to be sorted depending on the Boolean results.