Understanding

- I need to create a program called carLot2.cpp
  - It needs to define three classes
    - date
    - car
    - carlot
  - The main program loop should give the following options
    - add a car
    - print inventory
    - print profit for a given month/year
      - Dates must be validated, minus years and leap years

Design

```
Class Date
{
	//implementation and constructors
	//functions
	// - getters
	// - validdate()
	// - validMonthDayCombo
}

Class Car
{
	//implementation and constructors
	//functions
	// - getters
	// - getprofit()
}

Class CarLot
{
	//implementation and constructors
	//functions
	// - AddCar (see previous assignment for pseudocode)
	// - PrintInv (see previous assignment for pseudocode)
	// - CalculateProfit (see previous assignment for pseudocode)
}

int main()
{
	//Calls various class functions to achieve design goals
}
```

Testing

1 Chevy Volt 2014 25000 2 28 2014 n 2 3 2 2014	Details of car with N/A in sale related fields No cars sold in time period given	same
1 Chevy Volt 2014 25000 2 28 2014 n 1 Ford Ranger 2013 15000 2 30 // invalid date 2014 //loops 2 5 2014 y 20000 2 10 2014 2 3 2 2014	Says invalid date Cars in inventory 1 car sold 5000.00 profit exits	same

Reflection

I was able to copy and past a lot of code, which I guess is expected, because structs and classes are fairly similar. I can't see a lot of reason for not using classes (and just abandoning structs altogether), except for maintaining backwards compatibility with how C-programmers see things (structs can only contain POD).

Attaching functions to data structures seems to be a nice way to keep implementation close to the data. I see in the final project that we will get a chance to try splitting up implementation and definition (using header files); I think that will be interesting.

I don't think I really had any issues with the assignment. It was pretty straightforward, especially because I could copy and paste code.