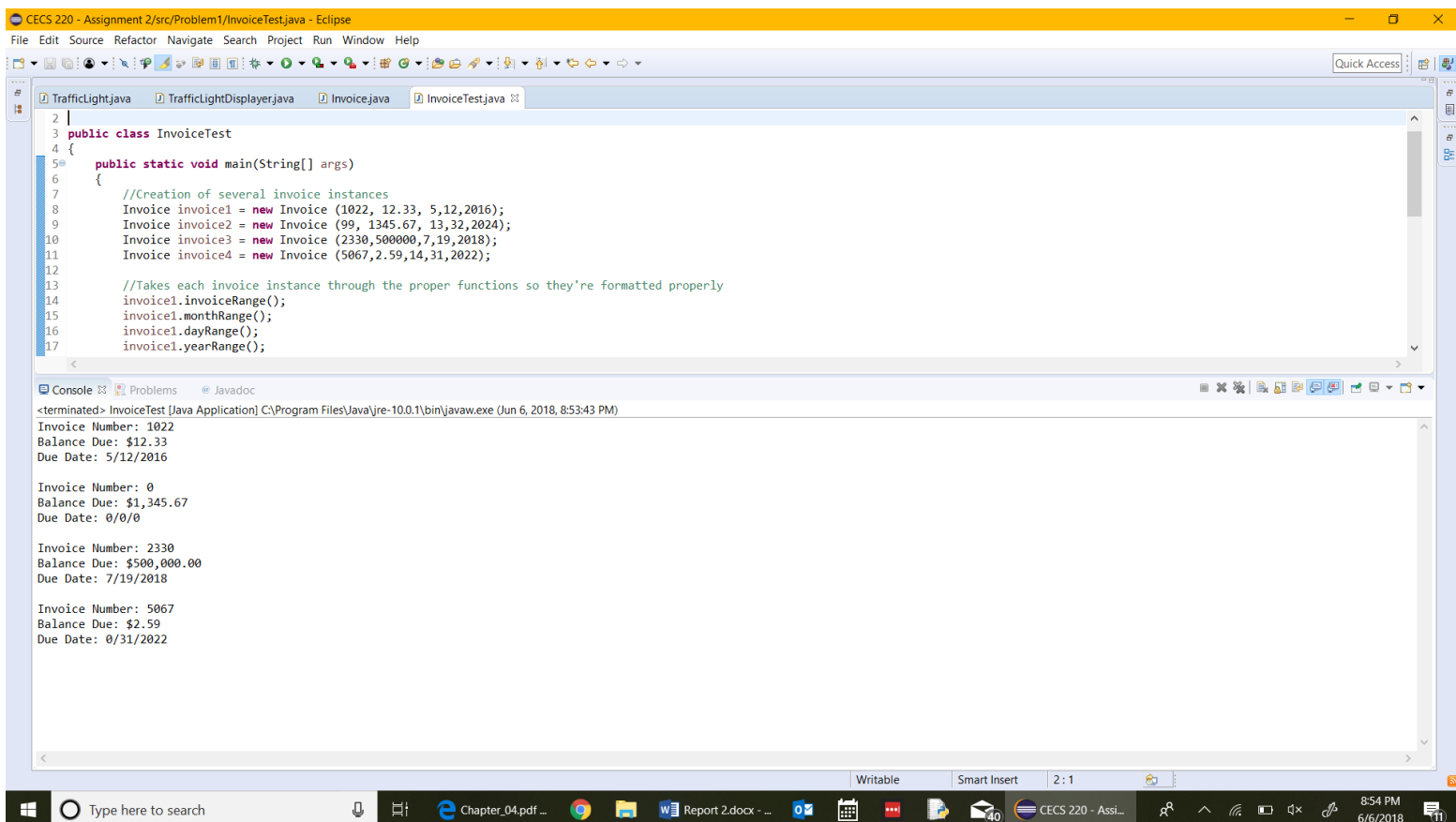


**Jeremy Boyd**  
**Assignment 02**  
**6/6/2018**

# Problem 1

This problem presented its challenges but overall wasn't too difficult, first I created my *Invoice* class and imported the *NumberFormat* class to be used later. Next, I created five private variables for the *invoiceNumber*, *balanceDue*, *month*, *day*, and *year*. Then I created the constructor for the class and created five variables that were named the same as my instance variables. I then used the *this* function so I could refer to the object whose method was being invoked. Next, I created my setters and getters for each variable so they could be retrieved and returned properly. Then I created methods which each value except the balance due could go through to make sure they were each within the proper parameters. And lastly, I created a *toString* method that would format the invoices in a neater, more visually appealing manner. I also formatted the *balanceDue* variable with the *getCurrencyInstance()* so it's format would look nicer. Next I created the *InvoiceTest* class and I created four instances from my invoice class, the numbers I chose had no significance I just tested values that were within range and some that were out of range. I put each of the four instances through the four methods to validate the values of the invoice number, month, day, and year. Then I set them to be displayed on the screen.



The screenshot shows the Eclipse IDE with the `InvoiceTest.java` file open. The code defines a `public class InvoiceTest` with a `main` method that creates four `Invoice` objects and calls their `invoiceRange`, `monthRange`, `dayRange`, and `yearRange` methods. The console output shows the results of these method calls for each invoice instance.

```
2 |
3 | public class InvoiceTest
4 | {
5 |     public static void main(String[] args)
6 |     {
7 |         //Creation of several invoice instances
8 |         Invoice invoice1 = new Invoice (1022, 12.33, 5,12,2016);
9 |         Invoice invoice2 = new Invoice (99, 1345.67, 13,32,2024);
10 |        Invoice invoice3 = new Invoice (2330,500000,7,19,2018);
11 |        Invoice invoice4 = new Invoice (5067,2.59,14,31,2022);
12 |
13 |        //Takes each invoice instance through the proper functions so they're formatted properly
14 |        invoice1.invoiceRange();
15 |        invoice1.monthRange();
16 |        invoice1.dayRange();
17 |        invoice1.yearRange();
18 |    }
19 | }
```

Console Output:

```
<terminated> InvoiceTest [Java Application] C:\Program Files\Java\jre-10.0.1\bin\javaw.exe (Jun 6, 2018, 8:53:43 PM)
Invoice Number: 1022
Balance Due: $12.33
Due Date: 5/12/2016

Invoice Number: 0
Balance Due: $1,345.67
Due Date: 0/0/0

Invoice Number: 2330
Balance Due: $500,000.00
Due Date: 7/19/2018

Invoice Number: 5067
Balance Due: $2.59
Due Date: 0/31/2022
```

## Problem 2

This GUI assignment presented an interesting challenge and I enjoyed working my way around it because the final result is pretty cool. To start though I created the *TrafficLight* class and extended it as a *Pane* class. I then imported the necessary classes that I would need to design the traffic light. Next, I created circle objects for the red light, yellow light, and green light and I created a rectangle for the background of the traffic light, they were all created as public static variables, so they could be manipulated as necessary. Next, I created a constructor for the *TrafficLight* class, within it I set the initial fill colors for the circles as red, grey, grey because the traffic light begins on stop. And then I set the background color of the traffic light to black, next I used the *getChildren()* method to add all the objects I created onto the plane so they could be displayed. I then created enumerated variables of STOP, CAUTION, and GO that would be the trigger words for how the lights would switch. Lastly, created a *SwitchState* method with the variable *pressedButton* as type *State* to represent my enumerated variables. I used three if statements that would check which button was pushed (stop, caution, or go) and change the color of the circles accordingly. After this I created the *TrafficLightDisplayer* class and imported the necessary classes so the program would be able to operate properly including importing the *TrafficLight* pane so I could access the design of my traffic light. Next I created three buttons that said, “Stop”, “Caution”, and “Go” to represent the three states of a traffic light. After this I created a grid pane called *diagram* that I would use to organize my traffic light and buttons the way I wanted. Then I created the flow pane that would group the three buttons horizontally together and positioned them in the center at the bottom of the window, I also set a small gap in between the buttons just so it would look better visually. Next I added both the traffic light that I imported from my *TrafficLight* class and the flow plane that contained my buttons into the grid pane that I had created. Again I set a small gap in between them for visual purposes, then I created my scene including my grid pane and set the background to white. And lastly I created a method that would implement the actions of *SwitchState* from my *TrafficLight* class using the *pushedButton* variable I created. And then back in my primary stage I created three statements that would connect my buttons to their proper actions when pushed. Using a using a lambda expression as my event handler to relate to my enumerated values.

