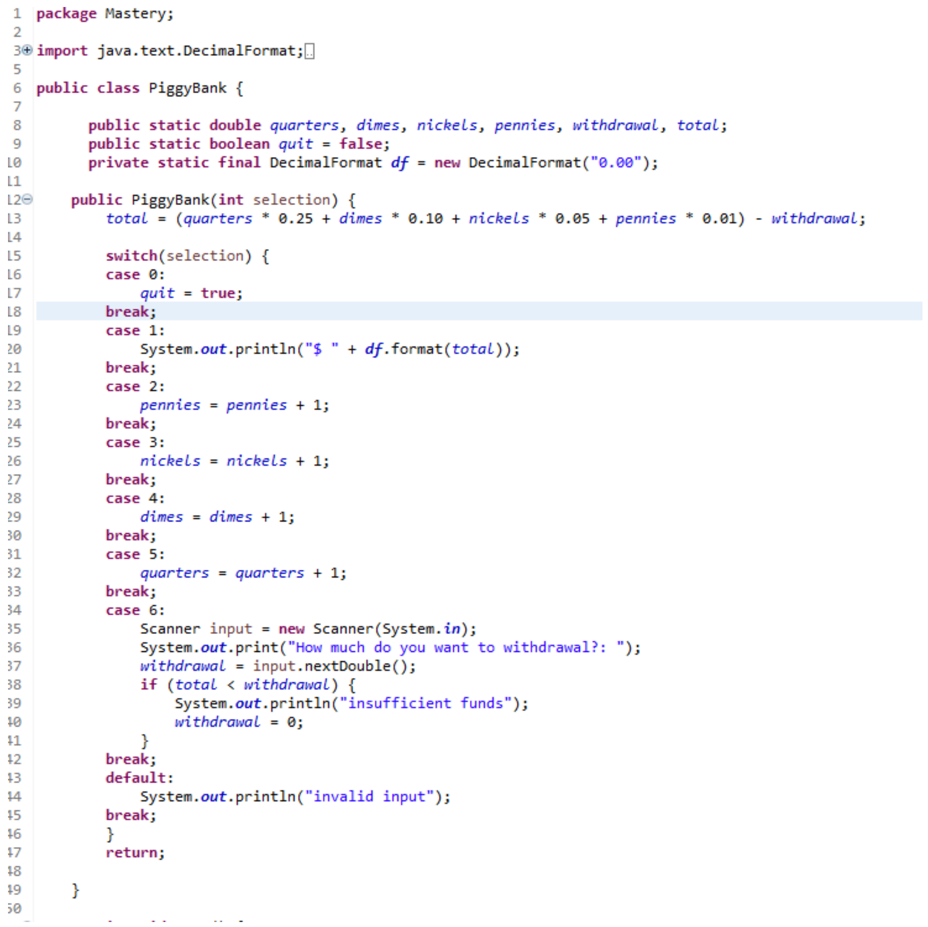
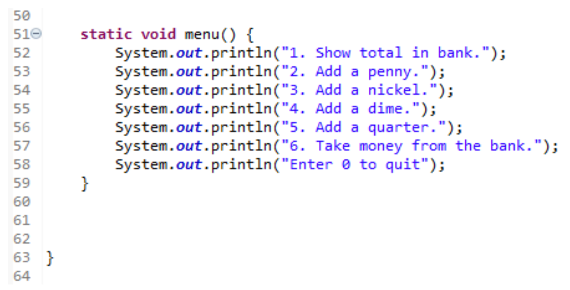
MySavings:
I was a bit confused at first because I had all the code in one java file but then I learned how to do it properly and I managed to make it as intended and with every input in mind. Such as printing ‘invalid input’ when you type anything other than the options on the menu or like when you try to withdraw more funds than you have it will print ‘insufficient funds’.

Piggybank code:

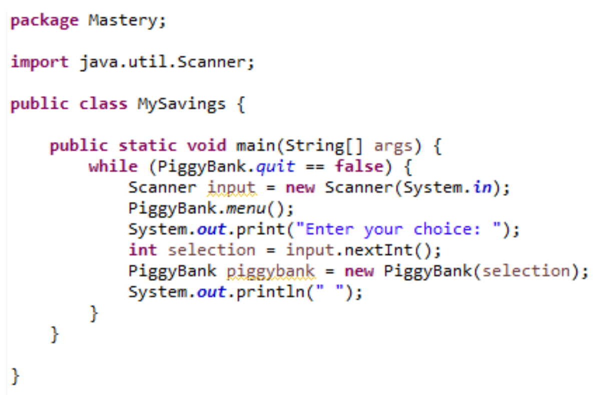


```java
package Mastery;

import java.text.DecimalFormat;

public class PiggyBank {

      public static double quarters, dimes, nickels, pennies, withdrawal, total;
      public static boolean quit = false;
      private static final DecimalFormat df = new DecimalFormat("0.00");

    public PiggyBank(int selection) {
        total = (quarters * 0.25 + dimes * 0.10 + nickels * 0.05 + pennies * 0.01) - withdrawal;

        switch(selection) {
        case 0:
            quit = true;
        break;
        case 1:
            System.out.println("$ " + df.format(total));
        break;
        case 2:
            pennies = pennies + 1;
        break;
        case 3:
            nickels = nickels + 1;
        break;
        case 4:
            dimes = dimes + 1;
        break;
        case 5:
            quarters = quarters + 1;
        break;
        case 6:
            Scanner input = new Scanner(System.in);
            System.out.print("How much do you want to withdrawal?: ");
            withdrawal = input.nextDouble();
            if (total < withdrawal) {
                System.out.println("insufficient funds");
                withdrawal = 0;
            }
        break;
        default:
            System.out.println("invalid input");
        break;
        }
        return;

    }

```



```java

    static void menu() {
        System.out.println("1. Show total in bank.");
        System.out.println("2. Add a penny.");
        System.out.println("3. Add a nickel.");
        System.out.println("4. Add a dime.");
        System.out.println("5. Add a quarter.");
        System.out.println("6. Take money from the bank.");
        System.out.println("Enter 0 to quit");
    }



}

```

MySavings code:

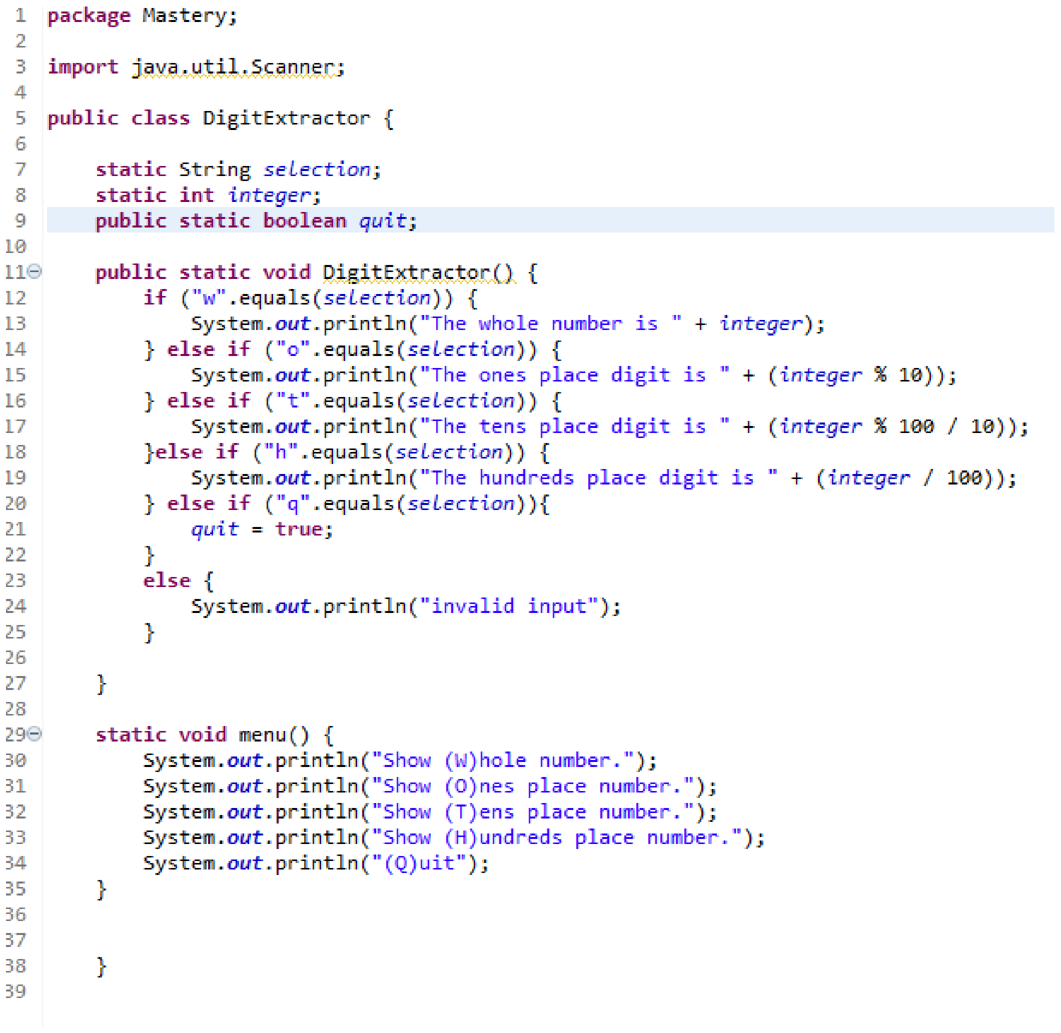


```
package Mastery;

import java.util.Scanner;

public class MySavings {

    public static void main(String[] args) {
        while (PiggyBank.quit == false) {
            Scanner input = new Scanner(System.in);
            PiggyBank.menu();
            System.out.print("Enter your choice: ");
            int selection = input.nextInt();
            PiggyBank piggybank = new PiggyBank(selection);
            System.out.println(" ");
        }
    }

}
```

<u>DigitExtractor:</u>

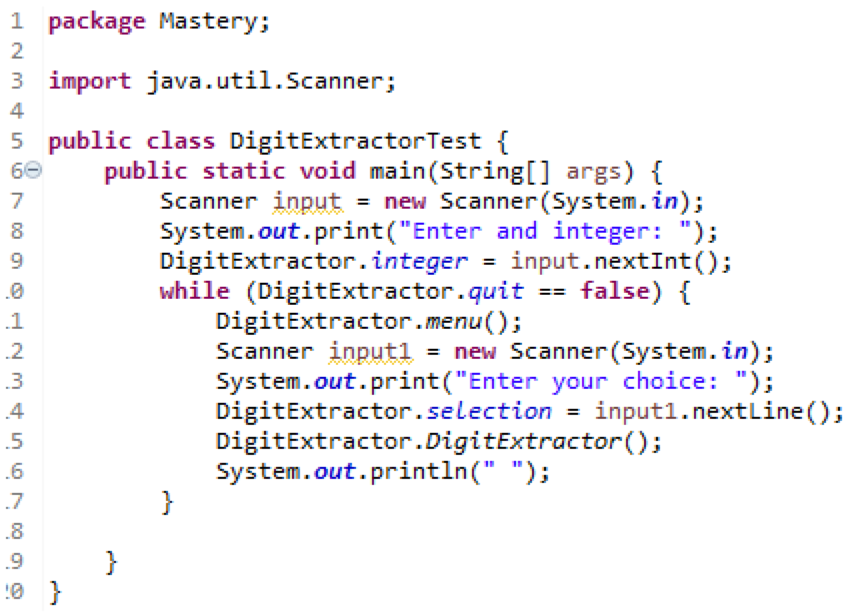
This was very reminiscent of the digits application in chapter 3 so I just used a lot of the code from that project and from MySavings

<u>DigitExtractor code:</u>



```
package Mastery;

import java.util.Scanner;

public class DigitExtractor {

    static String selection;
    static int integer;
    public static boolean quit;

    public static void DigitExtractor() {
        if ("w".equals(selection)) {
            System.out.println("The whole number is " + integer);
        } else if ("o".equals(selection)) {
            System.out.println("The ones place digit is " + (integer % 10));
        } else if ("t".equals(selection)) {
            System.out.println("The tens place digit is " + (integer % 100 / 10));
        }else if ("h".equals(selection)) {
            System.out.println("The hundreds place digit is " + (integer / 100));
        } else if ("q".equals(selection)){
            quit = true;
        }
        else {
            System.out.println("invalid input");
        }

    }

    static void menu() {
        System.out.println("Show (W)hole number.");
        System.out.println("Show (O)nes place number.");
        System.out.println("Show (T)ens place number.");
        System.out.println("Show (H)undreds place number.");
        System.out.println("(Q)uit");
    }


    }

```

<u>DigitExtractorTest code:</u>

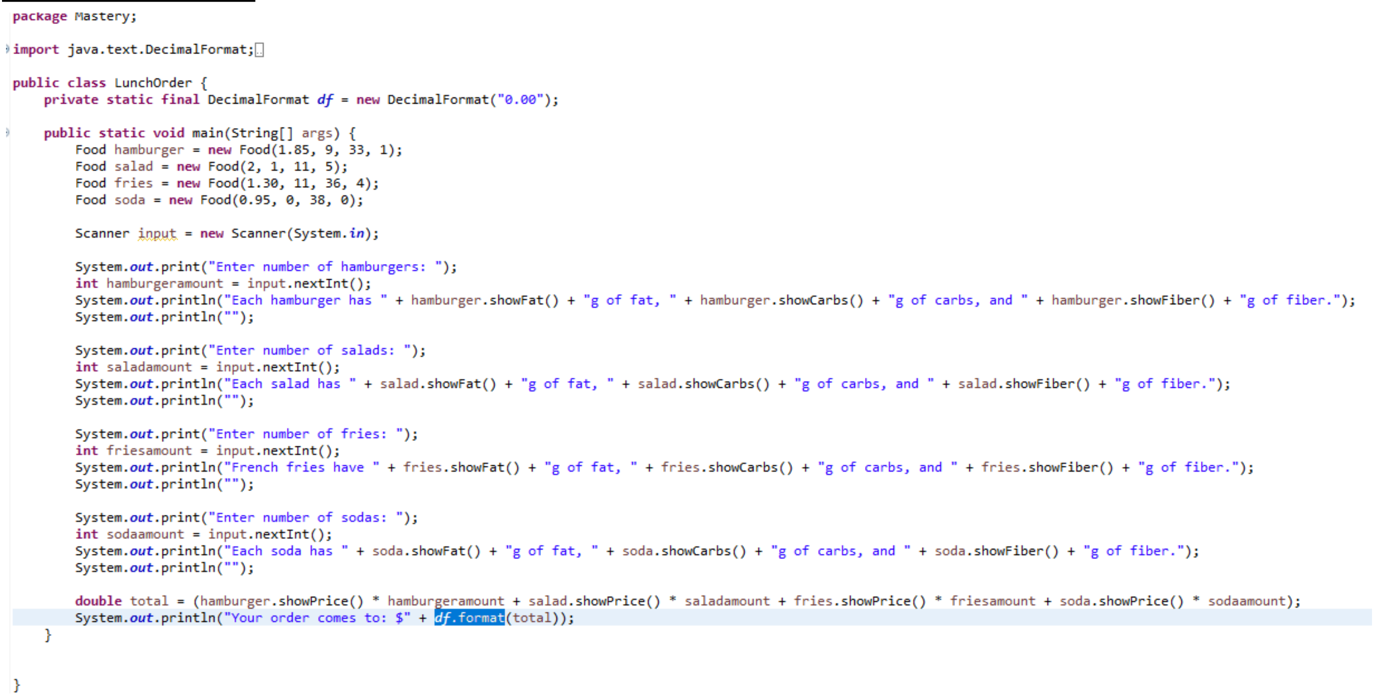


```java
package Mastery;

import java.util.Scanner;

public class DigitExtractorTest {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        System.out.print("Enter and integer: ");
        DigitExtractor.integer = input.nextInt();
        while (DigitExtractor.quit == false) {
            DigitExtractor.menu();
            Scanner input1 = new Scanner(System.in);
            System.out.print("Enter your choice: ");
            DigitExtractor.selection = input1.nextLine();
            DigitExtractor.DigitExtractor();
            System.out.println(" ");
        }

    }
}
```

LunchOrder:

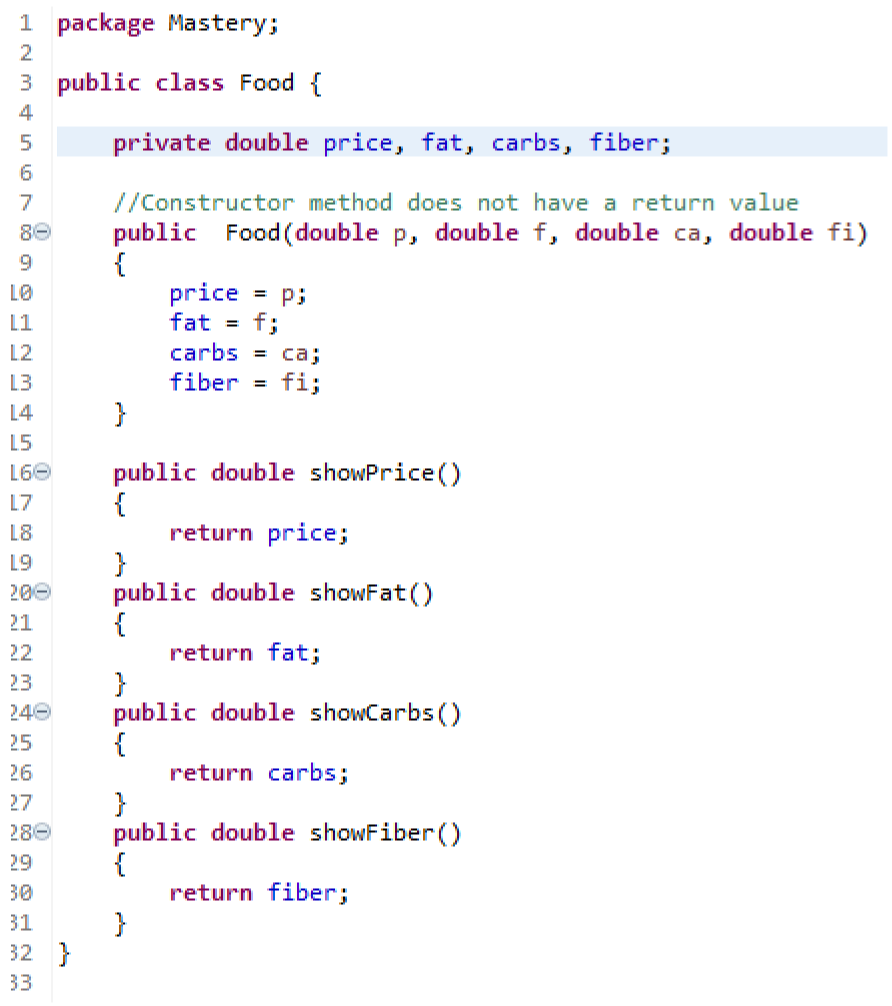
This was when I truly learned about object oriented programming and it started to all make sense to me. I was a bit stumped but once it was explained to me, it seemed so simple. I liked the fact that you can assign your own numbers for all the food items and that makes it easier and less bloated.

LunchOrder code:



```java
package Mastery;

import java.text.DecimalFormat;

public class LunchOrder {
    private static final DecimalFormat df = new DecimalFormat("0.00");

    public static void main(String[] args) {
        Food hamburger = new Food(1.85, 9, 33, 1);
        Food salad = new Food(2, 1, 11, 5);
        Food fries = new Food(1.30, 11, 36, 4);
        Food soda = new Food(0.95, 0, 38, 0);

        Scanner input = new Scanner(System.in);

        System.out.print("Enter number of hamburgers: ");
        int hamburgeramount = input.nextInt();
        System.out.println("Each hamburger has " + hamburger.showFat() + "g of fat, " + hamburger.showCarbs() + "g of carbs, and " + hamburger.showFiber() + "g of fiber.");
        System.out.println("");

        System.out.print("Enter number of salads: ");
        int saladamount = input.nextInt();
        System.out.println("Each salad has " + salad.showFat() + "g of fat, " + salad.showCarbs() + "g of carbs, and " + salad.showFiber() + "g of fiber.");
        System.out.println("");

        System.out.print("Enter number of fries: ");
        int friesamount = input.nextInt();
        System.out.println("French fries have " + fries.showFat() + "g of fat, " + fries.showCarbs() + "g of carbs, and " + fries.showFiber() + "g of fiber.");
        System.out.println("");

        System.out.print("Enter number of sodas: ");
        int sodaamount = input.nextInt();
        System.out.println("Each soda has " + soda.showFat() + "g of fat, " + soda.showCarbs() + "g of carbs, and " + soda.showFiber() + "g of fiber.");
        System.out.println("");

        double total = (hamburger.showPrice() * hamburgeramount + salad.showPrice() * saladamount + fries.showPrice() * friesamount + soda.showPrice() * sodaamount);
        System.out.println("Your order comes to: $" + df.format(total));
    }


}
```

Food code:



```java
package Mastery;

public class Food {

    private double price, fat, carbs, fiber;

    //Constructor method does not have a return value
    public  Food(double p, double f, double ca, double fi)
    {
        price = p;
        fat = f;
        carbs = ca;
        fiber = fi;
    }

    public double showPrice()
    {
        return price;
    }
    public double showFat()
    {
        return fat;
    }
    public double showCarbs()
    {
        return carbs;
    }
    public double showFiber()
    {
        return fiber;
    }
}

```