Mastery LunchOrder Reflection Log

Credit Name: CSE 3120 Object-Oriented Programming 1
Assignment Name: Mastery LunchOrder

How has your program changed from planning to coding to now? Please explain?

From planning to coding I took the UML diagram that I made and started to flush out each variable and method in their respective class (LunchOrder and Food) into code. Each part became more complex as I made a line of code for the various methods and variables spread out between the two classes. Adding the object also took much longer than the simple diagram in the UML diagram; however after watching some videos I figured out how to properly transfer information between classes, leading to a successful program.

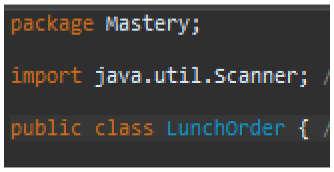
From coding to now I did not change any of the code besides the misspelled object in my LunchOrder class (explained in error log), along with this I completed the error log and the reflection log after finishing coding.

Steps

LunchOrder

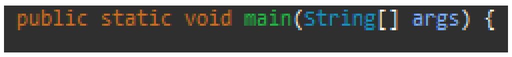
(the package this file is in is mastery)

I started off by importing java.util.scanner which allows me to get different data types from user inputs. The public class is declared as LunchOrder(It's declared as public so the PiggyBank class can access its information and return information).



```
package Mastery;

import java.util.Scanner;

public class LunchOrder {
```

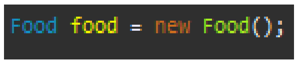
Below it shows the public (accessible from other classes) method "main" is static (you can access it without creating an object), void (method doesn't return any value), main (method name), and String[ ] args (String represents the data type [ ] indicates that it will be an array, and args (arguments) is the name of the array).



```
public static void main(String[] args) {
```

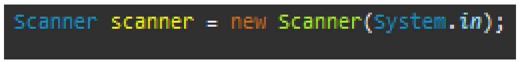
I then created the food object.
-Food is the type of object created (is the Food class which contains the methods and variables related to the user's bank).
-food is the new object that the PiggyBank class above is assigned to (Food classes values will become food objects values as well)
-new is used to create memory(space) to store the food object

-Food() this is a constructor that initializes the new object (filling the memory(space) with information for the object from the class Food)



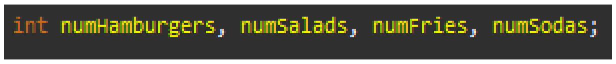
```
Food food = new Food();
```

The Scanner object is created below to take in inputs from the user. (I named mine scanner)



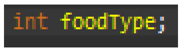
```
Scanner scanner = new Scanner(System.in);
```

I then declared my four items as integer variables. These aren't the items themselves but the number of each item the user wants to order (hence why I but num in front of each food type)



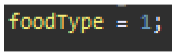
```
int numHamburgers, numSalads, numFries, numSodas;
```

Next, I declared the integer variable foodType. I do this to prepare this variable for each of the four item cases. This variable's value will be determined by what food it's on. Ex. hamburgers means foodType = 1, this is then sent to the Food class to be used in the case (in the switch statement) that will return the nutritional information.



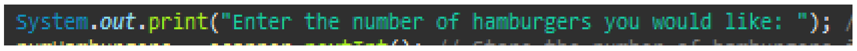
```
int foodType;
```

After this I made the foodType variable equal to 1 so that the Food class know's it should return the hamburgers nutritional information in the next print statements.



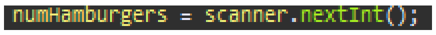
```
foodType = 1;
```

I then prompted the user to enter the number of hamburgers they would like.



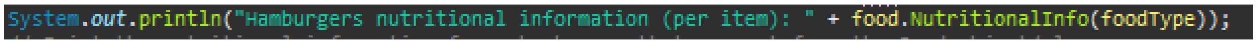
```
System.out.print("Enter the number of hamburgers you would like: "); /
```

Next, I saved this input in the variable numHumburgers that we declared before with the other foods num. (Note: this is already declared as an integer variable above so I don't need to write that)



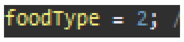
```
numHamburgers = scanner.nextInt();
```

After this I printed the nutritional information (per item) for hamburgers that I retrieved from the Food class method NutrtionalInfo. (this is explained more in the Food class reflection log)



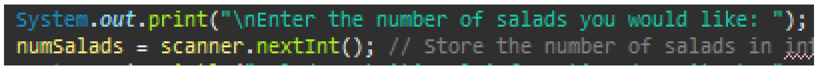
```
System.out.println("Hamburgers nutritional information (per item): " + food.NutritionalInfo(foodType));
```

I then made the foodType equal to 2 to indicate to the Food class that we're now doing the salad food item.



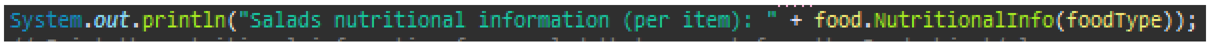
```
foodType = 2; /
```

Next, I made the prompt statement to ask the user for the number of salads they want, and then saved their input as the numSalads variable that we declared before. (Note: this is already declared as an integer variable above so I don't need to write that)



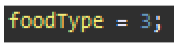
```
System.out.print("\nEnter the number of salads you would like: ");
numSalads = scanner.nextInt(); // Store the number of salads in int
```

After this I printed the nutritional information (per item) for salads that I retrieved from the Food class method NutrtionalInfo. (this is explained more in the Food class reflection log)



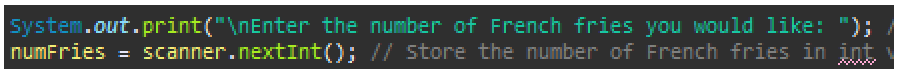
```
System.out.println("Salads nutritional information (per item): " + food.NutritionalInfo(foodType));
```

I then made the foodType equal to 3 to indicate to the Food class that we're now doing the French fries food item.



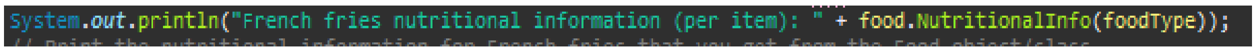
```
foodType = 3;
```

Next, I made the prompt statement to ask the user for the number of French fries they want, and then saved their input as the numFries variable that we declared before. (Note: this is already declared as an integer variable above so I don't need to write that)



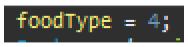
```
System.out.print("\nEnter the number of French fries you would like: "); /
numFries = scanner.nextInt(); // Store the number of French fries in int
```

After this I printed the nutritional information (per item) for French fries that I retrieved from the Food class method NutrtionalInfo. (this is explained more in the Food class reflection log)



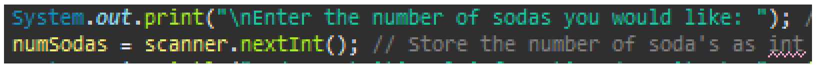
```
System.out.println("French fries nutritional information (per item): " + food.NutritionalInfo(foodType));
```

I then made the foodType equal to 4 to indicate to the Food class that we're now doing the soda's food item.



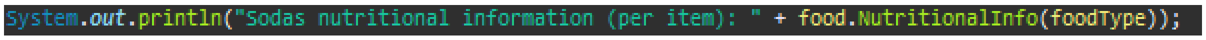
```
foodType = 4;
```

Next, I made the prompt statement to ask the user for the number of soda's they want, and then saved their input as the numSodas variable that we declared before. (Note: this is already declared as an integer variable above so I don't need to write that)



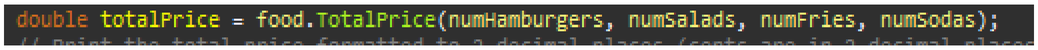
```
System.out.print("\nEnter the number of sodas you would like: "); /
numSodas = scanner.nextInt(); // Store the number of soda's as int
```

After this I printed the nutritional information (per item) for soda's that I retrieved from the Food class method NutrtionalInfo. (this is explained more in the Food class reflection log)



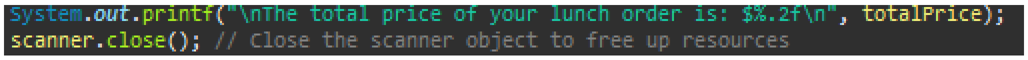
```
System.out.println("Sodas nutritional information (per item): " + food.NutritionalInfo(foodType));
```

I then made a double variable totalPrice and this was equal to the totalPrice variable that was calculated in the TotalPrice method in the Food class, that used the numHumburgers, numSalads, numFrieds, and numSodas values as its parameters.



```
double totalPrice = food.TotalPrice(numHamburgers, numSalads, numFries, numSodas);
```

Lastly, I printed a sentence indicating to the user what the total price of their order was. I print the totalPrice variable that I created above, and I format this to 2 decimal places because you don't usually write money with 1 decimal place. Ex. $47.10 instead of $47.1



```
System.out.printf("\nThe total price of your lunch order is: $%.2f\n", totalPrice);
scanner.close(); // Close the scanner object to free up resources
```

I then closed the main method with a brace.

```
}
```

And closed the LunchOrder class with a brace.

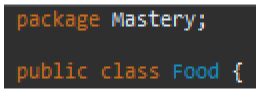
```
}
```

Together they looked like this.

```
    }
} //
```

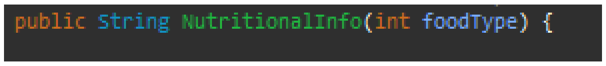
Food

(The package this file is in is mastery)
I start off by declaring the Food class as public so that the LunchOrder class can access and send information to it.



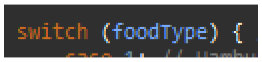
```
package Mastery;

public class Food {
```

Next, I began the NutrtionalInfo method, that is public and and string type so that it can return sentences to LunchOrder. I pass it the parameter foodType from LunchOrder so that in its switch statement it can determine which information/case to send back.



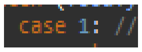
```
public String NutritionalInfo(int foodType) {
```

After this, I created the switch statement that would perform an action based on the value of foodType, hence why I passed it the foodType variable for the cases.



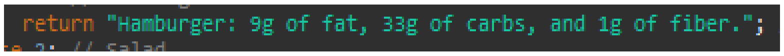
```
switch (foodType) {
```

I then created the first case which correlated to the food hamburgers.



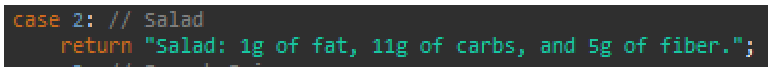
```
case 1: //
```

I made a return message (this is sent to LunchOrder) that when printed by LunchOrder would show the user the nutritional information of 1 hamburger.



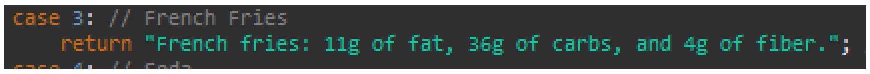
```
return "Hamburger: 9g of fat, 33g of carbs, and 1g of fiber.";
```

The next case created (case 2) was for salads. Once the user was finished entering the number of hamburgers in LunchOrder, the foodType variable would become equal to 2. This corresponds to this case which would then return the nutritional information of 1 salad for LunchOrder to print for the user.



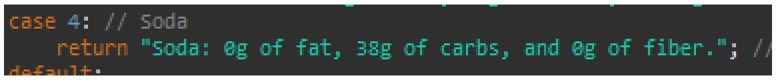
```
case 2: // Salad
    return "Salad: 1g of fat, 11g of carbs, and 5g of fiber.";
```

The next case created (case 3) was for French fries. Once the user was finished entering the number of salads in LunchOrder, the foodType variable would become equal to 3. This corresponds to this case which would then return the nutritional information of 1 French fries for LunchOrder to print for the user.



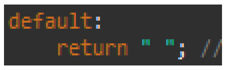
```
case 3: // French Fries
    return "French fries: 11g of fat, 36g of carbs, and 4g of fiber.";
```

The next case created (case 4) was for sodas. Once the user was finished entering the number of French fries in LunchOrder, the foodType variable would become equal to 4. This corresponds to this case which would then return the nutritional information of 1 soda for LunchOrder to print for the user.



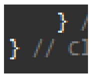
```
case 4: // Soda
    return "Soda: 0g of fat, 38g of carbs, and 0g of fiber."; //
```

The next case created was the default case which I simply made to return a space, because the default case would never happen. (the foodType variable is predetermined to a set(s) number in LunchOrder so you can't get a default case, since it will only equal 0-4 (0 because it's initialized as that) before the program stops)



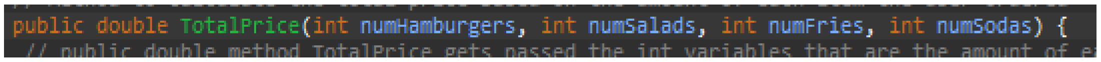
```
default:
    return " "; //
```

I then closed the switch statement and the NurtionalInfo method with these two brace.



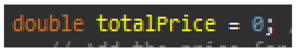
```
    } /
} // cl
```

I then made the public double method TotalPrice (It's double because it's working with decimals and returning a decimal number, and public so that LunchOrder can access its information). I passed this method the variables that track the number of each food the user wanted (Ex. numHamburgers). (I pass it these variables to calculate the total price with)



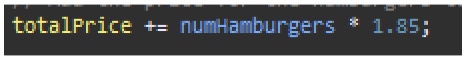
```
public double TotalPrice(int numHamburgers, int numSalads, int numFries, int numSodas) {
  // public double method TotalPrice gets passed the int variables that are the amount of e
```

I then declare a double variable totalPrice as 0 so that it can be used in calculations later in the method.



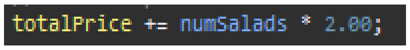
```
double totalPrice = 0;
```

The next line of code adds the number of hamburgers the user ordered times a hamburgers price ($1.85) to the totalPrice variable.



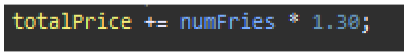
```
totalPrice += numHamburgers * 1.85;
```

Then I add the number of salads the user ordered times a salads price ($2.00) to the totalPrice variable.



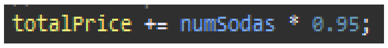
```
totalPrice += numSalads * 2.00;
```

Next, I add the number of French fries the user ordered times a French fries price ($2.00) to the totalPrice variable.



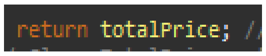
```
totalPrice += numFries * 1.30;
```

Lastly, I add the number of soda's the user ordered times a soda's price ($0.95) to the totalPrice variable.



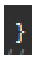
```
totalPrice += numSodas * 0.95;
```

I then return the totalPrice, which is now equal to all of the items's prices * number ordered combined.



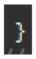
```
return totalPrice; /
```

Next I add the brace to close the TotalPrice method.



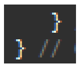
```
}
```

And a brace to close the public class Food.



```
}
```

Together they looked like this.



```
    }
} //
```