Ch. 8 CSE3130 - Object Oriented Programming 2: Error Log

Zephram Gilson

UEmployee, Faculty, Staff (With client code UniEmployee)

One of the minor errors I made in my coding process was forgetting to add the `super.` in the `super.toString()` method:

```
// Override toString to include department info
@Override
public String toString() {
    return toString() + ", Department: " + department;
}
```

As a result, my output looked like this:

```
    at mastery.UniEmployees.Staff.toString(Staff.java:40)
    at mastery.UniEmployees.Staff.toString(Staff.java:40)
    at mastery.UniEmployees.Staff.toString(Staff.java:40)
    at mastery.UniEmployees.Staff.toString(Staff.java:40)
    at mastery.UniEmployees.Staff.toString(Staff.java:40)
    at mastery.UniEmployees.Staff.toString(Staff.java:40)
    at mastery.UniEmployees.Staff.toString(Staff.java:40)
    at mastery.UniEmployees.Staff.toString(Staff.java:40)
    at mastery.UniEmployees.Staff.toString(Staff.java:40)
    at mastery.UniEmployees.Staff.toString(Staff.java:40)
    at mastery.UniEmployees.Staff.toString(Staff.java:40)
    at mastery.UniEmployees.Staff.toString(Staff.java:40)
    at mastery.UniEmployees.Staff.toString(Staff.java:40)
    at mastery.UniEmployees.Staff.toString(Staff.java:40)
    at mastery.UniEmployees.Staff.toString(Staff.java:40)
    at mastery.UniEmployees.Staff.toString(Staff.java:40)
```

It is necessary to add the **super**, as using `super.toString()` within the `toString()` method of the `Staff` subclass allows it to include details defined in the `UEmployee` superclass.

My final (modified) code looked like this:

```
    // Override toString to include job title info
    @Override
    public String toString() {
        return super.toString() + ", Job Title: " + jobTitle;
    }
}
```

This was the only major logic error I made in my coding process.

Vehicle, Car, Truck, Minivan (With client code CreateVehicle)

One of the errors I made in my code was when I implemented a method to every class to get and return its vehicle-specific attribute, for example in my **Minivan** class,

```
// Additional method specific to Minivan
public boolean hasRearEntertainmentSystem() {
    return hasRearEntertainmentSystem;
}
```

I encountered a logic error when I tried to call/use this method in the client code:

```
// Create a Car object
Car car = new Car(25.0, 35.0, 5, 15.0, "Sedan");
car.displayVehicleType();
System.out.println(car);

// Create a Truck object
Truck truck = new Truck(15.0, 20.0, 3, 30.0, 10000);
truck.displayVehicleType();
System.out.println(truck);

// Create a Minivan object
Minivan minivan = new Minivan(20.0, 28.0, 7, 40.0, true);
minivan.displayVehicleType();
minivan.hasRearEntertainmentSystem();
System.out.println(minivan);
```

I originally believed that this code would display the vehicle type, then call the method `hasRearEntertainmentSystem()` which was intended to return something along the lines of “True” based on the conditions of the Minivan object in this case.

My error here is that this line does not actually do anything, and that I have to implement it in something like an `if` statement, followed by more code like a `System.out.println()`

To show a proper output, I added an `if` statement to my client code:

```
        // Create a Minivan object
        Minivan minivan = new Minivan(20.0, 28.0, 7, 40.0, true);
        minivan.displayVehicleType();
        System.out.println(minivan);
        // Call the hasRearEntertainmentSystem method directly
        if (minivan.hasRearEntertainmentSystem()) {
            System.out.println("The minivan is equipped with a rear entertainment
system.");
        } else {
            System.out.println("The minivan does not have a rear entertainment
system.");
        }
```

Now, depending on the state of the boolean when a new **Minivan** (in this case)* is created, a respective output is printed.

* *this same logic could be applied to any additional vehicle, displaying the car's body type of the truck's towing capacity,*

This was the only major logic error I encountered in my coding process.

Account, PersonalAcct, BusinessAcct (With client code AccountsRun)

The first error I encountered was within my logic when testing my code:

```
Account ID: JDoe
John Doe
123 Main St, Springfield, IL 62704
Current balance is $150.00

Balance below minimum. Penalty of $2.0 applied.
Account ID: JDoe
John Doe
123 Main St, Springfield, IL 62704
Current balance is $88.00

Account ID: JSmith
Jane Smith
456 Elm St, Shelbyville, CA 62565
Current balance is $600.00

Balance below minimum. Penalty of $10.0 applied.
Account ID: JSmith
Jane Smith
456 Elm St, Shelbyville, CA 62565
```

The output seemed confusing at first because of two reasons:

1. If the minimum balance is $100 and $500 for personal and business accounts respectively, both accounts' balances are above these amounts and should not experience the penalty
2. The final balance after the penalty does not match up with what it should be.

The reason for this was because in my client code, I was trying to perform withdrawal functions on each account and print the accounts' information another time after the withdrawal:

```
public class AccountsRun {
    public static void main(String[] args) {
        // Test Personal Account with minimum balance penalty
        PersonalAcct personalAccount = new PersonalAcct(150.0, "John", "Doe", "123 Main St", "Springfield", "IL", "62704");
        System.out.println(personalAccount);
        personalAccount.withdrawal(60); // Should apply penalty
        System.out.println(personalAccount);

        // Test Business Account with minimum balance penalty
        BusinessAcct businessAccount = new BusinessAcct(600.0, "Jane", "Smith", "456 Elm St", "Shelbyville", "CA", "62565");
        System.out.println(businessAccount);
        businessAccount.withdrawal(200); // Should apply penalty
        System.out.println(businessAccount);
    }
}
```

Seeing this, the output makes sense now, as:

[personalAccount object]

150 (Initial balance) – 60 (Withdrawal) = 90; 90 – 2 (Penalty) = **88**

[businessAccount object]

600 (Initial balance) – 200 (Withdrawal) = 400; 400 – 10 (Penalty) = **390**

To make what the code is actually doing more evident in the output, I added a line to output a message letting the user know that money has been withdrawn from the account.

This error could have additionally been avoided entirely by excluding the withdrawal from the code.

However, to keep the withdrawal in the code, I modified my code to look like this:

```
public class AccountsRun {
    public static void main(String[] args) {
        // Test Personal Account with minimum balance penalty
        PersonalAcct personalAccount = new PersonalAcct(150.0, "John", "Doe", "123 Main St", "Springfield", "IL", "62704");
        System.out.println("=== Personal Account Details ===");
        System.out.println(personalAccount);

        // Test withdraw function
        System.out.println("\nWithdrawing $60 from Personal Account...");
        personalAccount.withdrawal(60); // Should apply penalty if below minimum balance
        System.out.println(personalAccount);

        // Test Business Account with minimum balance penalty
        BusinessAcct businessAccount = new BusinessAcct(600.0, "Jane", "Smith", "456 Elm St", "Shelbyville", "CA", "62565");
        System.out.println("\n=== Business Account Details ===");
        System.out.println(businessAccount);

        // Test withdraw function
        System.out.println("\nWithdrawing $200 from Business Account...");
        businessAccount.withdrawal(200); // Should apply penalty if below minimum balance
        System.out.println(businessAccount);
    }
}
```

```
=== Personal Account Details ===
Account ID: JDoe
John Doe
123 Main St, Springfield, IL 62704
Current balance is $150.00


Withdrawing $60 from Personal Account...
Balance below minimum. Penalty of $2.0 applied.
Account ID: JDoe
John Doe
123 Main St, Springfield, IL 62704
Current balance is $88.00


=== Business Account Details ===
Account ID: JSmith
Jane Smith
456 Elm St, Shelbyville, CA 62565
Current balance is $600.00


Withdrawing $200 from Business Account...
Balance below minimum. Penalty of $10.0 applied.
Account ID: JSmith
Jane Smith
456 Elm St, Shelbyville, CA 62565
Current balance is $390.00
```

Now, the output is more clear about what the program is doing.

This was the only major logic error I encountered in my coding process.