**Refactored Code**

**class** **NullCustomer** **extends** Customer {

**boolean** isNull() {

**return** **true**;

}

Plan getPlan() {

**return** **new** NullPlan();

}

}

*// Replace null values with Null-object.*

customer = (order.customer != **null**) ?

order.customer : **new** NullCustomer();

*// Use Null-object as if it's normal subclass.*

plan = customer.getPlan();

The conditional that deals with null has been implemented as a null object. In this case, the NullCustomer subclass inherits the Customer class. When a new customer joins, the program can simply execute the default methods in the NullCustomer subclass. If the customer is not null (existing customer) the relevant code can be executed.

**Testing**

Ensure the class returns the same functionality as the conditional before the refactoring. This can be done by checking both the null and non-null functionality to ensure they return the appropriate data.