Designing for testability

1. When is a class controllable? How do you make a class controllable?

2. When is a class observable? How do you make a class observable?

3. The book says: *whenever we need a spy to assert the behavior, we must ask ourselves why we need a spy*. Why is this?

4. Is the change shown in Listing 7.11 in the book a good solution to improve observability?

5. Is this testable? If not, why not?

```java
class InvoiceFilter {
   public List<Invoice> lowValueInvoices() {
       DatabaseConnection dbConnection = new DatabaseConnection();
       IssuedInvoices issuedInvoices = new IssuedInvoices(dbConnection);

       List<Invoice> all = issuedInvoices.all();
       return all.stream()
               .filter(invoice -> invoice.getValue() < 100)
               .collect(toList());
   }
}
```

6. Would you create a Clock class and create an attribute of type Clock in ChristmasDiscount? Or pass a value of type LocalDate to the applyDiscount method? Why?

```java
public class ChristmasDiscount {
   public double applyDiscount(double amount) {
       LocalDate today = LocalDate.now();
       double discountPercentage = 0;
       boolean isChristmas = today.getMonth() == Month.DECEMBER
               && today.getDayOfMonth() == 25;
       if (isChristmas)
           discountPercentage = 0.15;
       return amount - (amount * discountPercentage);
   }
}
```

7. How can you improve the testability of the following OrderDeliveryBatch class?

```java
public class OrderDeliveryBatch {
    public void runBatch() {
        OrderDao dao = new OrderDao();
        DeliveryStartProcess delivery = new DeliveryStartProcess();
        List<Order> orders = dao.paidButNotDelivered();
        for (Order order : orders) {
          delivery.start(order);
          if (order.isInternational()) {
            order.setDeliveryDate("5 days from now");
          } else {
            order.setDeliveryDate("2 days from now");
          }
        }
    }
}
class OrderDao {
  // accesses a database
}
class DeliveryStartProcess {
  // communicates with a third-party web service
}
```

8. How can you improve the testability of the following KingsDayDiscount class?

```java
public class KingsDayDiscount {
  public double discount(double value) {
    Calendar today = Calendar.getInstance();
    boolean isKingsDay = today.get(MONTH) == Calendar.APRIL
                              && today.get(DAY_OF_MONTH) == 27;
    return isKingsDay ? value * 0.15 : 0;
  }
}
```