Core C# Structure Lab

In these labs we'll be creating the infrastructure of an ecommerce website for the venerable Northwind Traders, a company that sells wholesale, high-end gourmet foods to restaurants and cafes.

1. Find a lab partner. Find someone you don't regularly work with and ideally someone with a different skillset than yourself.

2. Make sure that at least one of your machines has Visual Studio 2015 or better, Entity Framework 4.1 or better, and MSTest. If not, go ahead and get them installed.

3. Create a NEW solution called Labs. It should start off with a Windows Forms Application called WindowsBackOffice.

4. Look around a bit. Explore the project. See what files and resources are included.

5. In that solution you'll have one project that was auto-created. Go ahead and add a new project to the solution called "Core". It should be a Class Library project. (Hint: right-click on the Solution and choose "Add New Project").

6. In Core, create these business classes. Each class should be in a separate Class file:
   - Category
   - Customer
   - Employee
   - EmployeeTerritory
   - OrderDetail
   - Order
   - Product
   - Region
   - Supplier
   - Territory

7. Ask your lovely and talented instructor for a database. Get attached to it and look through the tables. Any of those look familiar? They look like your business classes!

8. Begin to add properties to your business classes based on the columns in your tables. Use the same names. Use C# data types as you create the properties recognizing that database data types are not the same as C# data types. In other words, you'll need to do some translations.

9. Open your OrderDetail.cs class. Add this property:

```
public virtual Product Product { get; set; }
```

   This is an example of how a class can *have* an object. It is called *aggregation* in OO lingo.

10. Build your project and resolve any issues you may have.

No, it isn't very exciting yet, but hang in there. We'll add some excitement soon!