Day 11: The abstract factory method

Take the factory method, and instead of creating new classes for particular implementations (we had a `PgSqlClient`) and then creating instances of that, we abstract one level. Now we can have instances corresponding to particular implementations, rather than classes! So like this:

```scala
class DatabaseClient(connectorFactory: DatabaseConnectorFactory) {
  def executeQuery(query: String): Unit = {
    val connection = connectorFactory.connect()
    connection.executeQuery(query)
  }
}

val clientMySql: DatabaseClient = new DatabaseClient(new MySqlFactory)
val clientPgSql: DatabaseClient = new DatabaseClient(new PgSqlFactory)
```

This is:

- Very easy to inject mocks
- Easy to add new instances and refactor
- Hides details of instantiation, much like the normal factory method