Section 9 Welcome Message

This material starts with an essential and beautiful contrast in how you can decompose many programming problems either by using multiple functions each with a set of cases (like in an ML case expression) or by using multiple classes each with a set of methods -- and that these "entirely opposite" approaches are essentially a different choice in how to arrange your code, as well as what subsequent additions to the code do not require modifying code already written. Understanding how both approaches "work" and relate to each other is a great way to understand both of them better.

We then build on this key idea in a couple ways. First, we consider situations where functional decomposition "works fine" but sticking with an object-oriented approach requires a more complicated idiom called *double-dispatch*. Second, we consider *multiple inheritance*, which takes the distinguishing feature of object-oriented programming and makes it more powerful by allowing for multiple superclasses. Because doing so leads to problems as well as benefits, we then consider approaches that have some of the benefits and fewer of the problems, considering Ruby's *mixins* and Java's *interfaces* (no knowledge of Java is needed).

Overall, this module has less video content but a more challenging and rewarding homework that brings together several topics, both from this week and from previous ones.

✓ **Completed**

Go to next item