Coordinate has

boolean equals(Coordinate c2)

boolean equals(ColorCoordinate c2)

ColorCoordinate has

boolean equals(Coordinate sample)

boolean equals(ColorCoordinate sample)

There's different rule for which class's methods get invoked versus what happens with the parameters.

A Coordinate reference can point to a ColorCoordinate object, BUT

A ColorCoordinate reference can NOT point to a Coordinate object.

WITH the object a method is called on, POLYMORPHISM occurs and Java is smart and detects what class the object a method was called on and uses the methods from the object class not the reference class.

BUT WITH PARAMETERS, JAVA goes with the reference class. For it to go with the object class, you must CAST the reference to the right class.

On line #70 in ColorCoordinate, when we print test, due to POLYMORPHISM, Java detects that even though the reference type of test is Coordinate, the OBJECT it's pointing to is a ColorCoordinate and the toString method in ColorCoordinate is invoked.

nocolor.equals(red) calls a method in Coordinate because nocolor is referencing a Coordinate object. It's going to call Method #2 because red is a ColorCoordinate reference.

red.equals(nocolor) calls a method ColorCoordinate because red is referencing a ColorCoordinate object. It's method 3 because the reference type of nocolor is Coordinate.

red.equals(blue) calls a method in ColorCoordinate because red is referencing a ColorCoordinate object and blue is a ColorCoordinate reference.

test.equals(blue) calls a method in the ColorCoordinate class because test is referencing a ColorCoordinate object. It's method #4 since blue is a ColorCoordinate reference.

blue.equals(test) calls a method in ColorCoordinate because blue is referencing a ColorCoordinate object. It calls method #3 since test is a Coordinate reference, but still returns false since the colors don't match.

Note

I talked through the whole PacMan example without writing anything down, sorry! Take some time to watch the video and simultaneously parse through the code to see why certain things went in certain places. Notice that the inheritance bit was quite seamless. Notice how move just sort of worked, how equal just sort of worked, because both were built into base classes...