With group 12

Linear Depreciation

Good
Having two constructors for both bigdecimal and double are useful and help with the class' use as BigDecimal is unwieldy.
Using chronounit is good example of code reuse rather than making their own methods to deal with it

To Improve
Could have Javadoc to help with coding outside of the class
Could have a few small comments to explain the two different constructors
Use Validation checks to ensure that null has not been passed and that the depreciation rate makes sense (0<x<1), this is a good use of defensive programming.
Could use assertions during testing.

Double Declining

Good
Two constructors are good again as this means the user can avoid dealing with BigDecimal
Reuse of the chronounit code is good as it saves time and fits in with the projects wide use of this style of date.

To Improve
Again some javadoc or comments to explain the wider ideas would be useful
Use Validation checks to ensure that null has not been passed and that the depreciation rate makes sense (0<x<1), this is a good use of defensive programming.
Use assertions during testing

Tests
Good
Tests multiple cases and objects

To Improve
Remove or resolve the TODOs in the tests as this is somewhat confusing
Wider range of dates or depreciation rate to check more possible outcomes which might cause the code to fail

General Comments
The code is generally good and of a high quality. However it could be improved with a bit of polishing including more comments/javadoc and some validation or assertions.