1. **Extension submodules**

* Implementation of extension submodule: 10%

Two class Linear Depreciation and Double Declining Balance Depreciation class has implemented required method in Valuation Policy correctly. The calculation is done in BigDecimal. The Test for calculation of both policies were implemented. Test passed.

* Peer review of other group’s submodule: Bonus 10%

Peer review of another group’s submodule was also completed. We feedbacked and discussed good point and the point can be improved each other.

1. **Tests**

* System tests covering key use cases: 18%

In the System Test, there exist a test to check each of the user case. In addition, there is a integrated test which test which integrate all three use cases as one flow. It also tests integration of submodules. Many data are prepared for System Test. In the test, use MockDeliveryService which is returned by DelivaryServiceFactory.

* Unit tests for Location and DateRange: 5%

Unit test for Location and DateRange class is written. They test main functionality of the class isNear() for the Location, and overlap() for the DateRange. For the test of these function, there are more than one test with different data.

* Systems test including implemented extension to pricing/valuation: 5%

The Integration Test in system test checks using extension valuation policy in use case, or one scenario.

* Mock and test pricing/valuation behavior given other extension **(challenging):** 3%

Valuation Policy is used in the integration test in system test, and checked whether it worked in side the scenario.

**3. Code**

* Integration with pricing and valuation policies: 10%

The Valuation policy is correctly implemented and can be used as part of system. In addition, without these Policy, system has default calculation method.

* Functionality and correctness: 25%

All the class needed for use cases required is constructed, and each of the class has necessary method for these use cases. The correctness of these class and method are proved in the system test which simulate all the required use case.

* Quality of design and implementation: 3%

The structure of the system is kept in simple, as well as class implementation. Method is implemented in appropriate class in the system. There are also some error catching “if” statement in the code.

* Readability: 5%

There are not TAB use in code, and for the Location and DateRange class, there are class-level, method-level, and field-level Javadoc.

**4. Report**

* Revisions to design: 5%

Class diagram, sequence diagram and communication diagram are updated to appropriate structure. Discussion is done for these re-designing.

* Self–assessment: 5%

Self-assessment is done with enough justification.