M226b Grading Criteria for larger Task Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

|  |  |  |  |
| --- | --- | --- | --- |
|  | (max) | **Points** | |
| **Implementation**  **Your own project has been implemented…**  with use cases (diagram, short descriptions)  with a sequence diagram  with class diagrams  (a rough sketch at beginning, detailed at end of project)  class diagrams are generated  **Code is runnable**  should run, but doesn’t   it runs for certain use cases, with others it crashes  runs fully  **Code is refactored** No redundancies in code   switch-cases are only used when it makes sense    **Exception-Handling is used**  own Exceptions implemented   **Input and output** via console  using files / data source  user friendly and handles wrong input  **JavaDoc** is used  for min. 80% of classes and min. 80% of methods  **JUnit** is applied  for min. 2 classes with min. 4 test methods  for a higher test coverage  Mockings are used when necessary | **2 P  + 1 P + 2 P + 2 P -2 P** | 7 |  |
| **- 2 P 1P + 1 P** | 9 |  |
| **1 P**  **+ 1 P + 1 P** | 12 |  |
| **1 P + 1 P** | 14 |  |
| **1 P + 1 P + 2 P** | 18 |  |
| **1 P** | 19 |  |
| **1 P + 2 P + 1 P** | 23 |  |
| **Object-orientated concepts in code**  OO-concepts are shown with own code  Encapsulation / Information Hiding is applied  Inheritance is being used Composition is implemented if useful Classes are well structured (ie. what they need to do with clearly defined methods)  Clean implementation of abstract classes / interfaces (or it is explained why this is not implemented)  Project is presented to teacher | **2 P** | 25 |  |
| **1 P 1 P 1 P + 1 P** | 29 |  |
| **3 P** | 32 |  |
| **2 P** | 34 |  |
| **Documentation** covering concept & design, planning and test protocol (use cases) | **4 P** | 38 |  |

Maximum points: 38; for grade 6 you need min.34pts.; for grade 5: min.27pts; for grade 4: min.20pts; for grade 3 min.12pts. If you have below 25pts, the competence-grid is reference for final grade.