Observations (GRADE)

1. Grade Grade

Attribute studentId:

- Diagram: studentId : String
- Code: private String studentId;
- Remark: There is no discrepancy. The name and type of the attribute match between diagram and code.

Attribute subjectCode:

- Diagram: subjectCode : String
- Code: private String subjectCode;
- Remark: No discrepancy. Attribute name and type match between diagram and code.

Attribute grade:

- Diagram: grade : double
- Code: private double grade;
- Remark: No discrepancy. The name and type of the attribute match between the diagram and the code.

- **Uncommunicative Names**
  The names studentId , subjectCode and grade are clear and communicative.
- **Inconsistent Names**
  Names are consistent throughout the code.
- **Long Methods**
  Methods are short and specific.
- **Duplicate Code -There is no duplicate code in the code.**
  There is no duplicate code in the Grade class.
- **Class Explosion**
  There is no sign of a class explosion. The Grade class is concrete and specific.
- **Large Classes**
  The Grade class is small and focused on a single responsibility.
- **Conditional Complexity**
  There is no complex conditional logic in the class.
- **Redundant or Meaningless Comments**
  Initial comments automatically generated by NetBeans do not add value to the code.
  Recommendation: Remove automatic comments generated by NetBeans and add meaningful comments where necessary.
- **Dead Code**
  There is no dead code in the Grade class.
- **Speculative Generality**
  The Grade class shows no signs of speculative generality.

Additional Recommendations

- **Use of Gson Annotations:**

Expose annotations are used correctly.
Recommendation: verify that all properties to be serialised or deserialised are correctly annotated.

- **Data Validation:**
The grade value is not being validated, which could allow values outside an expected range (e.g. negative grades or greater than the maximum allowed value).

Observations (Subject)

2. Class Subject

Attribute code:

- Diagram: code : String
- Code: private String code;
- Remark: There is no discrepancy. The name and type of the attribute match between diagram and code.

Attribute name:

- Diagram: name : String
- Code: private String name;
- Remark: No discrepancy. The name and type of the attribute match between the diagram and the code.

Attribute credits:

- Diagram: credits : int
- Code: private int credits;
- Remark: No discrepancy. The name and type of the attribute match between the diagram and the code.

- **Uncommunicative Names**
Code, name and credits names are clear and communicative.
- **Inconsistent Names**
Names are consistent throughout the code.
- **Long Methods**
Methods are short and specific.
- **Duplicate Code**
There is no duplicate code in the Subject class.
- **Class Explosion**
There is no sign of a class explosion. The Subject class is concrete and specific.
- **Large Classes**
The Subject class is small and focused on a single responsibility.
- **Conditional Complexity**
There is no complex conditional logic in the class.

- **Redundant or Meaningless Comments**
  Initial comments automatically generated by NetBeans do not add value to the code.
  Recommendation: Remove the automatic comments generated by NetBeans and add meaningful comments where necessary.
- **Dead Code**
  There is no dead code in the Subject class.
- **Speculative Generality**
  The Subject class shows no signs of speculative generality.

Additional Recommendations

- **Use of Gson Annotations**
  The @Expose annotations are used correctly.
  Recommendation: Verify that all properties to be serialised or deserialised are correctly annotated.
- **Data Validation:**
  The value of credits is not being validated, which could allow values outside an expected range (e.g. negative credits).
  Recommendation: Implement validations to ensure that credits values are valid.

Observations (IndividualActivity)

3. Class IndividualActivity

Attribute grade:

- Diagram: Attribute name is not explicitly specified, but grade : double is assumed.
- Code: private double grade;
- Remark: There is no discrepancy. The name and type of the attribute match the expectation based on the implicit diagram.

- **Uncommunicative Names**
  The name of the IndividualActivity class and the name of the grade attribute are clear and communicative.
- **Inconsistent Names**
  Names are consistent throughout the code.
- **Long Methods**
  Methods are short and specific.
- **Duplicate Code**
  There is no duplicate code in the IndividualActivity class.
- **Class Explosion**
  There is no sign of a class explosion. The IndividualActivity class is concrete and specific.
- **Large Classes**

The IndividualActivity class is small and focused on a single responsibility.

- **Conditional Complexity**
  There is no complex conditional logic in the class.

- **Redundant or Meaningless Comments**
  Initial comments automatically generated by NetBeans do not add value to the code.
  Recommendation: Remove the automatic comments generated by NetBeans and add meaningful comments where necessary.
- **Dead Code**
  There is no dead code in the IndividualActivity class.
- **Speculative Generality**
  The IndividualActivity class shows no signs of speculative generality.

Additional Recommendations

- **Data Validation:**
  The grade value is not being validated, which could allow values outside an expected range (e.g. negative grades or above the maximum allowed).
  Recommendation: Implement validations to ensure that grade values are valid.

Observations (QuimestralExam)

4. QuimestralExam class

Attribute grade:

- ➢ Diagram: Attribute name is not explicitly specified, but grade : double is assumed.
- ➢ Code: private double grade;
- ➢ Remark: There is no discrepancy. The name and type of the attribute match the expectation based on the implicit diagram.

- **Uncommunicative Names**
  The name of the QuimestralExam class and the name of the grade attribute are clear and communicative.
- **Inconsistent Names**
  Names are consistent throughout the code.
- **Long Methods**
  Methods are short and specific.
- **Duplicate Code**
  There is no duplicate code in the QuarterlyExam class.
- **Class Explosion**

There is no sign of a class explosion. The QuarterlyExam class is concrete and specific.

- **Large Classes**
  The QuarterlyExam class is small and focused on a single responsibility.
- **Conditional Complexity**
  There is no complex conditional logic in the class.
- **Redundant or Meaningless Comments**
  Initial comments automatically generated by NetBeans do not add value to the code.
  Recommendation: Remove automatic comments generated by NetBeans and add meaningful comments where necessary.
- **Dead Code**
  There is no dead code in the QuarterlyExam class.
- **Speculative Generality**
  The QuarterlyExam class shows no signs of speculative generality.

Additional Recommendations

- **Data validation:**
  The grade value is not being validated, which could allow values outside an expected range (e.g. negative grades or above the maximum allowed).

Recommendation: Implement validations to ensure that grade values are valid.