

BERUFSSKOLLEG WIRTSCHAFTSINFORMATIK

2

Musterklassenarbeit OOP

Name	Klasse	Datum	Punkte
			___ / 70

Bearbeitungszeit: 120 Minuten · Gesamtpunktzahl: 70 · Arbeitsfassung v1.2

Hilfsmittel: Entwicklungsumgebung und Structogrammer nach schulischer Festlegung. KI-Systeme nur, wenn die Lehrkraft dies ausdrücklich erlaubt.

Ausgangssituation

Ein Volleyballverein verwaltet Teams und Mitglieder. Daten werden über ein Repository gespeichert. Die Oberfläche soll austauschbar bleiben; Beitragsarten werden polymorph berechnet und Zugriffe abgesichert.

Aufgabe 1: Objektmodell und Beziehungen (15 P)

Entwickeln Sie ein UML-Klassendiagramm für Team, Mitglied und Beitragspflicht. Verwenden Sie Kapselung, Multiplizitäten und mindestens ein Interface. Begründen Sie Komposition oder Aggregation anhand der Lebensdauer.

Aufgabe 2: Polymorphe Beitragsberechnung (14 P)

Implementieren Sie die Anlage `Beitragspflicht.java` für `ErwachsenenMitglied` und `JugendMitglied`. Der aufrufende Code darf weder `instanceof` noch eine Typabfrage verwenden. Ergänzen Sie einen Testaufruf für beide Varianten.

Aufgabe 3: MVC und Fehlerweg (10 P)

Ordnen Sie View, Controller, Service und Repository ihren Verantwortungen zu. Beschreiben Sie den Ablauf eines Klicks auf „Aufnehmen“ einschließlich Validierungsfehler. SQL darf weder in View noch Controller erscheinen.

Aufgabe 4: JDBC-Repository vervollständigen (16 P)

Vervollständigen Sie `JdbcMitgliedRepository.java`: `findById` und `insert` mit `PreparedStatement`, `try-with-resources`, Parameterbindung und `ResultSet`-Mapping. `findById` liefert `Optional.empty()`, wenn kein Datensatz vorhanden ist.

Aufgabe 5: Sichere Software (10 P)

Analysieren Sie den in `UnsichererLogin.java` vorgegebenen SQL-String. Erklären Sie das Risiko und korrigieren Sie den Zugriff mit einem `PreparedStatement`. Unterscheiden Sie Passwort-Hashing und Verschlüsselung an je einem passenden Anwendungsfall.

Aufgabe 6: Qualität und Refactoring (5 P)

Nennen und konkretisieren Sie zwei automatisierte Tests und einen Refactoring-Schritt, mit denen die Erweiterbarkeit des Systems nachgewiesen wird. Jede Maßnahme benötigt einen beobachtbaren Nachweis.

Anlagen: Beitragspflicht.java • JdbcMitgliedRepository.java • Mitglied.java • UnsichererLogin.java • schema.sql