Programmieren, Design und Implementieren von Algorithmen (Programmierprojekt)

Gruppengröße: 4–5 Personen

Die Prüfungsleistung umfasst:

- Schriftliche Ausarbeitung
- Programm (inkl. Quellcode, Dokumentation, JavaDoc)
- Präsentation

Schriftliche Ausarbeitung

Gewichtung: 60 % der Gesamtnote, gemeinsam mit dem Programm

Formale Vorgaben:

- Umfang: 2.300–2.700 Wörter (ohne Inhalts-, Abbildungs- und Quellenverzeichnis sowie Fußnoten und Anhang). Codebeispiele gehören in den Anhang und zählen nicht um Umfang.
- Abgabeform: gedruckt (z. B. mit Klemmschiene) am Tag der Präsentation sowie elektronisch (TraiNex, siehe Frist online).
- Form und Stil: entsprechend der Richtlichen zum wissenschaftlichen Schreiben
- Nutzen Sie das Deckblatt für Seminararbeiten.
- Zielgruppe: Leser mit Java-Kenntnissen auf Niveau der Lehrveranstaltung.

Gliederung der Ausarbeitung:

- **Problembeschreibung**
    - Projektname und Projektlogo.
    - Beschreibung des Szenarios bzw. der Problemstellung in eigenen Worten.
    - Teamübersicht: Wer war für welche Teile der Programmierung, Dokumentation und Präsentation verantwortlich? Dokumentieren Sie auch die Verteilung der Arbeitszeit je Teammitglied.
- **Teamorganisation und Tools**

- Beschreibung der Zusammenarbeit im Team:
  - Rollenverteilung
  - Kommunikationswege und Entscheidungsfindung
  - Meetingstruktur und Absprachen
- **Einsatz von Tools:**
  - Nutzung von Versionskontrolle (z. B. Git / GitHub / GitLab)
  - Projektmanagement mit z. B. Kanban

- **Darstellung von Lösungsalternativen**
  Beschreibung möglicher konzeptioneller und technischer Lösungswege.
  - Mindestens ein übergeordnetes Thema (z. B. Klassenhierarchie, Architekturansatz)
  - Mindestens eine konkrete Detailentscheidung (z. B. Datentyp, Algorithmus, Speicherstrategie).
- **Begründete Auswahl einer Lösung**
  - Erläuterung, warum die gewählte Lösung den Alternativen vorgezogen wurde.
- **Beschreibung des Programms**
  - Übersicht der Hauptkomponenten und ihrer Funktionsweise (kein Code)
  - Beschreibung ausgewählter zentraler Klassen und ihrer Rollen.
  - Einsatz eines UML-Klassendiagramms im Anhang.
- **Beispielhafte Programmnutzung aus Anwendersicht**
  - Simulieren Sie eine typische Nutzung durch einen Anwender.
  - Verwenden Sie Screenshots, um typische Abläufe zu visualisieren.
- **Fazit und Ausblick**
  - Kurze Zusammenfassung.
  - Vorschläge für mögliche Erweiterungen oder Verbesserungen des Programms.
- **Zitate und Quellenverzeichnis**
  - Angabe relevanter Quellen (bspw. zu Algorithmen oder Dokumentation von Klassen/Bibliotheken).
  - Vollständig und korrekt formatiertes Literaturverzeichnis.

Anforderungen an das Programm

Modularität (Klassen, Methoden, Vererbung, Polymorphismus)

- Das Programm soll modular und objektorientiert aufgebaut sein. Achten Sie auf die Einhaltung der Grundprinzipien der objektorientierten Programmierung (z. B. Kapselung, Abstraktion, Vererbung, Polymorphie).
- Klassenstruktur:
    - Jede Klasse soll genau eine zentrale Verantwortung erfüllen (Single-Responsibility-Prinzip).
    - Vermeiden Sie zu große „God Classes“ mit zu vielen Aufgaben.
    - Trennen Sie klar zwischen Logik, Datenhaltung und ggf. Benutzerinteraktion.
- Methodenstruktur:
    - Methoden sollten eine klar definierte Aufgabe erfüllen und möglichst kurz und wiederverwendbar sein.
    - Vermeiden Sie Duplikationen und implementieren Sie wiederkehrende Funktionalität in Hilfsmethoden.
- Die Klassen sollen möglichst lose gekoppelt um spätere Erweiterungen oder Änderungen zu erleichtern.
- Wenn möglich, verwenden Sie Interfaces oder Abstraktionen, um die Architektur flexibel zu gestalten (z. B. für unterschiedliche Implementierungen oder zum Testen).
- Nutzen Sie das Werfen und Fangen von Exceptions, wo anwendbar.

Dokumentation und Codequalität

- JavaDoc ist Pflicht und umfasst:
    - Alle Konstruktoren, alle öffentlichen Methoden
        - Mit **@param** für Parameter
        - Mit **@return** für Rückgabewerte (außer bei void)
    - Ausnahmen: Klassen und Methoden der grafischen Benutzeroberfläche sind von JavaDoc-Pflicht ausgenommen.
    - Zusätzlich zum JavaDoc sind erklärende Kommentare im Code bei:
        - Komplexen Schleifen, Nicht-trivialen Bedingungen (if, switch)
        - Berechnungen oder Logik, die nicht selbsterklärend ist
- Einheitliche Sprachwahl (Deutsch oder Englisch) innerhalb Javadoc

Präsentation

Gewichtung: 40 % der Gesamtnote. Es gelten alle formalen Anforderungen an Präsentationsleistungen der BA.

Dauer: 20–30 Minuten

Handout: Reichen Sie zur Präsentation ein ein- bis zweiseitiges Handout ein, das die zentralen Inhalte knapp zusammenfasst.

Inhalte der Präsentation:

- Folgen Sie inhaltlich der Gliederung Ihrer schriftlichen Ausarbeitung. Erläutern Sie dabei die Problemstellung, die Teamorganisation, Lösungsalternativen und Entscheidungen, Funktionsweise und Besonderheiten Ihres Programms. Zeigen Sie den Einsatz Ihres Programms im Rahmen einer anwenderorientierten Live-Demo.
- Beschreiben Sie nicht zeilenweise den Code, sondern konzentrieren Sie sich auf übergeordnete Zusammenhänge (z. B. Architektur, Anwendungslogik, besondere Herausforderungen).

Live-Demo:

- Im Verlauf oder am Ende der Präsentation ist das entwickelte Programm live vorzuführen. Die Live-Demo sollte inhaltlich vorbereitet sein und zeigt genau zwei Funktionalitäten des Programms.
- Machen Sie keine unvorbereiteten „Spontanbeispiele“. Überlegen Sie sich praxisnahe Inhalte/Abläufe, die Sie präsentieren.
- Testen Sie vorab die technische Kompatibilität des Präsentations-Laptops mit dem Projektor.

Medieneinsatz:

- Die Hauptpräsentation erfolgt per Bildschirmpräsentation (z. B. PowerPoint) und Live-Demo.
- Sie dürfen auch zusätzliche analoge Medien einsetzen (z. B. Whiteboard, Flipchart, Pinwand).

Elektronische Abgabe

- Abgabe erfolgt als **ZIP-Archiv** über TraiNex. Das Archiv muss enthalten:
    - Den vollständigen **Quellcode** (src, geordnet, kompilierbar in Eclipse)
    - Die **JavaDoc-Dokumentation** (javadoc, auch im HTML-Format)
    - Die **schriftliche Ausarbeitung** (report, PDF)
    - Die **Präsentation** (presentation, PDF)
    - Ggf. **verwendete externe Bibliotheken** (libs, sofern der Speicherplatz es erlaubt)
    - Strukturieren Sie das Archiv, sodass es in Eclipse importierbar und ausführbar ist (Ordner: src, javadoc, libs, report, presentation). Testen Sie dies.