Informationen zur Prüfung im Modul „Programmieren 2“ im WiSe 2024/2025 bei Nadja Krümmel

1. Ablauf der Prüfung

Die Prüfung im Modul „Programmieren 2“ besteht aus zwei Teilen:

Teil 1: Programmierprojekt

- Das Programmierprojekt muss gemäß den Abgabekriterien bis zum **28. Januar 2025** fertiggestellt und auf Moodle hochgeladen sein.
- Nach der Abgabe muss das Projekt innerhalb der darauffolgenden sieben Tage in einem Präsentationstermin vorgestellt werden.
- Hinweis: Falls das Projekt nicht abgegeben wird, wird dieser Prüfungsteil mit 0 Punkten bewertet.

Teil 2: Klausur

- Die Klausur findet im regulären Prüfungszeitraum des Fachbereichs MNI der Technischen Hochschule Mittelhessen (THM) statt.
  Hierzu ist der reguläre Weg zur Prüfungsanmeldung zu beachten.

Die Gesamtnote der Prüfungsleistung ergibt sich aus dem Durchschnitt der Bewertungen der beiden Prüfungsteile, wobei jeder Prüfungsteil mit mindestens 50% bestanden sein muss. Die Prüfungsleistung gilt als bestanden, wenn die Gesamtnote mindestens 50% ist.

2. Prüfung Teil 1 - Abgabekriterien

Der erste Teil der Prüfung besteht aus einem selbstgewählten Entwicklungsprojekt.

Die Prüfungsform ist angelehnt an die Prüfung im SoSe 2024 bei Dominikus Herzberg. Sie können Sie unter Verwendung des Live View Programming eine freie Idee als Programmierprojekt umsetzen.

2.1 Zwei Wege zum Entwicklungsprojekt:

A – LVP erweitern - einen eigenen, neuen View kreieren
Sie erweitern das LVP mit einem Clerk um eine View, d.h. Sie binden eine JavaScript-Bibliothek für eine View im Browser ein, schreiben einen Java-Wrapper zur Nutzung der Bibliothek und nutzen diese neue View, um eine interessante Anwendung zu realisieren.

Ein Beispiel: Zum Canvas-Element (ein HTML-Element im Browser) gibt es umfangreiche Möglichkeiten, 2D- oder 3D-Zeichnungen zu erstellen (siehe das Canvas-API). Schauen Sie sich im LVP den Turtle-Clerk an, der nur auf wenigen Operationen in einer Canvas basiert. Warum nicht eine Turtle erstellen, die in einem dreidimensionalen Raum umherläuft? (Natürlich nicht mit einem Punkt, wie in 2D, der dann zu Strichen wird, sondern mit einem Strich, der Flächen erzeugt.) Oder eine andere Idee, mit der Sie im Raum einfache geometrische Formen zu räumlichen Objekten zusammensetzen können. Schreiben Sie dazu dann eine interessante Anwendung: Wie wäre es mit einem ganz simplem Mini-Minecraft?

Wenn Ihnen das zu schwer ist oder zu kompliziert erscheint, schauen Sie sich die Ideen im nächsten Abschnitt an. Andererseits: Wenn Sie die Mühe einer neuen View auf sich nehmen, bekommen Sie 10 Bonuspunkte (d.h. 10% von 100 maximal erreichbaren Punkten für die Entwicklungsarbeit).

B – Sie entwickeln eine Anwendung, die LVP nutzt

Sie entwickeln eine Anwendung, die die bestehenden Views nutzt (vorrangig also Markdown, HTML und die Turtle). Hier finden Sie ein paar Beispiele:

- Nachbau einer grundlegenden, universellen Anwendung wie z.B. Excel oder einen Texteditor; hier geht es um die Entwicklung eines funktionalen Kerns, der das Potenzial hat, zu einer umfassenderen Anwendung ausgebaut zu werden. Entscheidend sind hier gut gewählte Datenstrukturen.
- Entwicklung eines Taschenrechners, der - und das ist die Herausforderung - bei der Implementierung keine Zahlentypen verwenden darf, weder explizit, noch implizit. Damit scheidet z.B. die Adressierung von Arrays über den Index aus oder der Längenvergleich zweier Zeichenketten. Es bietet sich an, einen Taschenrechner umzusetzen, der die umgekehrte polnische Notation verwendet.
- Entwicklung eines Simulators für digitale Schaltungen mit einem etwas komplizierteren Anwendungsbeispiel
- Entwicklung eines Funktionsplotters, mit dem Sie mathematische Ausdrücke wie z.B. $x2$oder sin(x) darstellen können; damit das anspruchsvoll wird, sollte man sehr leicht in den Graphen hinein und aus ihn herauszoomen können, den Anzeigebereich wählen können und die Performanz im Blick behalten.
- Schreiben Sie einen Markdown-Parser, der Markdown in HTML umwandelt (dafür werden Sie vermutlich mit regulären Ausdrücken arbeiten).
- Oder setzen Sie eine einfache Variante zur Visualisierung von Graphen (das sind Bilder aus Knoten und Kanten, meist Kreisen und Pfeilen) gemäß eines einfachen Layout-Verfahrens um, das dann mit Hilfe einer Turtle-Grafik gezeichnet wird. So etwas geht mit der DOT-Sprache. Sie müssen aber keine Sprache dafür entwickeln oder lesen können, sondern „nur“ eine Datenstruktur zur Abbildung von Graphen in ein geeignetes Arrangement zur Darstellung als Abbildung umrechnen.

2.2 Rahmenbedingungen

Bei beiden Varianten ist folgende Grundidee nicht aus dem Auge zu verlieren: Ihre Anwendung soll grundsätzlich über die JShell, d.h. über den Aufruf von Methoden steuerbar sein, was dann in der Regel zur Aktualisierung der Live View führt, wenn sich damit ein für die Darstellung relevanter Zustand verändert hat. Es geht also **nicht** vorrangig um eine **interaktive** View! Die Interaktivität ist eine indirekte über die JShell!

Angestrebte Ziele

- Sie zeigen, dass Sie moderne Programmierkonzepte von Java einsetzen, wie etwa Listen, Maps etc., funktionale Interfaces und Lambda Ausdrücke, Streams, Records samt Pattern Matching, Sealed Classes und Sealed Interfaces, String Templates usw.
- Testen Sie Ihre Programme. Nur vollständig lauffähige Projekte werden bewertet. Rechnen Sie genügend Zeit für einen umfangreichen Funktionstest ein.

2.3 Bewertungsschema

Werden die Anforderungen erfüllt? (50 P)
Sie legen fest, was das Programm leisten soll, und erläutern dies konkret anhand von exakt fünf ausgewählten Szenarien. Sie haben so die Möglichkeit den Umfang, den Schwierigkeitsgrad und damit die zu erreichende Punktzahl zu beeinflussen
Ihr Programm besteht aus einer einzigen Java-Datei, die sowohl den entwickelten Code als auch die LVP-Dokumentation enthält. Beim Programmablauf wird die Dokumentation im Browser erstellt und es werden die Live-Views zu den fünf Szenarien generiert. Jedes Szenario wird in einem eigenen Unterkapitel vorgestellt und als View demonstriert. Wählen Sie die Szenarien so, dass deutlich wird, was Ihr Programm kann und wie gut es z.B. einen oder zwei Sonder- oder Problemfälle beherrscht.
Das Funktionsversprechen und eine Übersicht zu den fünf Szenarien liefern Sie als PDF-Datei **Montag, den 2. Dezember 2024** in Moodle ab.

Beachten Sie formale Rahmenbedingungen? (30 P)

- ***Halten Sie die Kodierrichtlinien für Java ein (10 P)***
  Orientieren Sie sich an dem Kodierstil und den Namenskonventionen aus der Veranstaltung. Im Zweifel gelten die Google-Richtlinien und die Coding Standards von SE-EDU. Vollkommen unakzeptabel sind willkürlich eingerückte Code-Strukturen und eklatante Verletzungen der Schreibweisen für Bezeichner.
- ***Verwenden Sie moderne Sprachkonstrukte von Java (10 P)***
  Darunter fällt u.a. Folgendes: Interfaces, Collections, generische Programmierung, Streams, Lambda-Ausdrücke, Funktionsobjekte, Records, sealed Classes bzw. Interfaces, Pattern-Matching mit switch, Immutabilität, ein Programmierstil mit fluent interfaces, Immutabilität.
- ***Methodenrümpfe sind nicht zu lang (10 P)***
  Ein Methodenrumpf darf nicht mehr als 20 Semikolons beinhalten. Damit wird einerseits eine Dekomposition eines Programms in mehrere übersichtlichen Methoden gefördert und es gibt einen Anreiz, strombasierte Programmierung mit Streams umzusetzen.

Lesbarkeit/Verständlichkeit der LVP-Dokumentation (20 P)
Ist die LVP-Dokumentation lesbar und verständlich, ist der Code übersichtlich geschrieben und nachvollziehbar?

Wie anspruchsvoll ist die Herausforderung? (Gewichtungsfaktor)
Die Bewertung wird gewichtet anhand eines Faktors, der die Qualität der Abgabe einstuft.

- Faktor 0: Das Programm ist zu kurz, die Aufgabenstellung zu anspruchslos, es zeigen sich Fehler und Probleme
- Faktor 0,3: Der Schwierigkeitsgrad ist nicht angemessen, die Umsetzung problematisch, die Ergebnisse mangelhaft, der Code in Aufbau und Struktur ungenügend
- Faktor 0,5: Der Schwierigkeitsgrad ist zu gering, die Umsetzung noch akzeptabel, das Ergebnis befriedigend, der Code vom Aufbau und der Struktur problematisch
- Faktor 0,8: Der Schwierigkeitsgrad ist angemessen, die Umsetzung gelungen, das Ergebnis gut, der Code vom Aufbau und der Struktur gut nachvollziehbar

- Faktor 0,9: Wie Faktor 0,8, allerdings werden mindestens ein Kriterium von Faktor 1.0 erfüllt.
- Faktor 1.0: Der Schwierigkeitsgrad ist hoch, die Umsetzung anspruchsvoll, das Ergebnis tadellos, der Code ist vom Aufbau und der Struktur sehr gut nachvollziehbar

Punktbewertung

$$p=(a+f+l)\times g+b$$

p - Erreichte Punkte
a – Anforderungen
f – Formales
l – Lesbarkeit
g – Gewichtung
b - Bonus (10 Punkte)

Abgabemodalitäten

Sie geben Ihr Programm als gepackte zip-Datei zusammen mit dem LVP-Verzeichnis ab. Ihr Programm muss in der mitgelieferten Umgebung ausführbar sein, vorrangig in der JShell oder, wenn es besondere Gründe dafür gibt, als mit dem Kommando java ausführbares Programm. Sollte es nur mit dem Kommando java ausführbar sein, besprechen Sie dies vorher mit uns.

Die Abgabe ist am **28. Januar 2025** fällig. Die Dateien müssen bis zum Abgabeende in Moodle hochgeladen werden. Eine Verlängerung ist nur im Krankheitsfall möglich. Sollten Sie den Abgabetermin aus gesundheitlichen Gründen nicht einhalten können, melden Sie sich umgehend per E-Mail beim mir. Im Krankheitsfall weisen Sie Ihre Erkrankung mit einem Attest nach und erhalten maximal eine Verlängerung für die Abgabe über die Dauer ihrer Krankschreibung.

Vorgabe zum Funktionsversprechen

Ihr Projekt wird an Ihrem Funktionsversprechen und den Szenarien gemessen, die das Funktionsversprechen konkretisieren.

Gliedern Sie das Funktionsversprechen wie folgt:

- Titel zu Ihrer Entwicklungsarbeit mit Name und Matrikelnummer; geben Sie Ihrem Projekt einen Namen, der einen Hinweis gibt, worum es bei Ihnen geht
- Verwenden Sie die in Moodle verlinkte Vorlage. Sollten Sie ein anderes Format als .docx benötigen, sprechen Sie mich an.
- Funktionalität: Sie beschreiben, was der Zweck Ihrer Anwendung ist und worin die programmiertechnische Herausforderung besteht, die Sie mit dieser Entwicklungsarbeit lösen möchten.
- Achten Sie auf rechtschriftliche und grammatikalische Richtigkeit.
- 1. Szenario — Nummerieren Sie die Szenarien durch und geben Sie jedem Szenario einen Titel
- 2. Szenario usw. — Geben Sie exakt fünf Szenarien an

Zu den Szenarien:

- Beschreiben Sie, was für einen Ablauf, Teilschritt oder Anwendungsfall Sie mit diesem Szenario veranschaulichen und demonstrieren wollen.
- Skizzieren Sie bildlich (erstellen Sie eine Zeichnung oder scannen Sie eine Handskizze ein), wie die dazugehörige View aussieht. In manchen Fällen bietet sich eine Vorher/Nachher-View an.
- Erklären Sie, was in der View zu sehen ist.

In der LVP-Dokumentation zu Ihrem Projekt müssen Sie auf das Funktionsversprechen und die Szenarien eingehen und die Views der Szenarien erzeugen. Wenn Ihre Umsetzung vom Funktionsversprechen abweicht, weisen Sie darauf hin und dokumentieren Sie die Gründe ausführlich.

Abgabemodalitäten
Das Funktionsversprechen laden Sie in Moodle als PDF hoch.
Mit dem Hochladen des Funktionsversprechens sind Sie verbindlich zu diesem Teil der Prüfungsleistung angemeldet.

3. Prüfung Teil 2 – Klausur

Die Klausur wird in der Prüfungsphase es WiSe 2024 stattfinden. Sie umfasst den kompletten Teil Stoffumfang der Vorlesung/Übung und des Praktikums. Es wird erwartet, dass der Programmcode sich in Art und Form an den in der Veranstaltung verwendeten Konventionen orientiert. Schreiben Sie syntaktisch korrekten Code.

4. Termine für die Gesamtprüfungsleistung

VW	Was
7	28./29.11. Pflichtpraktikum
8	2.12. Abgabe Funktionsversprechen in Moodle 05./06.12. Pflichtpraktikum
9	12./13.12. freies Praktikum 13.12. Abgabe Teilnahmebestätigung
10	19./20.12. Pflichtpraktikum
Weihnachtsferien	
11	Freies Praktikum ausschließlich nach Anmeldung, Termin werden noch bekannt gegeben
12	16./17.01. Pflichtpraktikum
13	23./24.01. freies Praktikum
14	28.01. Abgabe Projekt in Moodle Do/Fr Projektpräsentation - Termineinwahl folgt