FH CAMPUS WIEN
UNIVERSITY OF APPLIED SCIENCES

Grundlagen der Programmierung
Projektvorschläge

Projektabgabe:
Bei der Projektabgabe sind insgesamt **25 Punkte** (von 100 Punkten) zu erreichen. 10 Punkte entfallen auf die Abgabe des Projekttagebuchs und der Meetingprotokolle, 5 Punkte auf die Peer Reviews und 10 Punkte auf die Applikation selbst sowie die Endpräsenation. Es ist erforderlich 15 Punkte (60%) zu erreichen, um die LV positiv abzuschließen. Hierbei handelt es sich um eine Gruppenarbeit und als solche wird die Gruppe als Ganzes benotet. Sollten Mitglieder einer Gruppe sich an der Arbeit nicht beteiligen, informieren Sie den jeweiligen Vortragenden.

Beurteilungskriterien Programmcode:

- Programm ist lauffähig und stürzt bei gewöhnlicher Eingabe/Bedienung nicht ab,
- Source Code Verständnis ist bei allen Gruppenmitgliedern vorhanden.
- Code Stil: „optisch ansprechende Umsetzung / Formatierung" / „Liebe zum Detail"
- Ist das Programm verständlich strukturiert? Klassenwahl. Methodeneinsatz...
- Source Code Dokumentation: Kommentare zu Klassen und Methoden vorhanden (nicht jede Zeile kommentieren!)

Projektvorschläge

Angabe von Aufwand/Komplexität:
(1=niedrig, 3=mittel, 5=hoch)
Bitte wählen Sie eine für das Team angemessene Komplexität.

Es können auch **eigene Projektvorschläge** eingebracht werden. Diese sind in Hinblick auf Umfang mit den KollegInnen und Lektoren abzustimmen.

Die Projekte werden in **GitHub Repositories** verwaltet. Die Lektoren sind als Projektmitglieder hinzuzufügen, wenn die Projekte privat eingestellt werden.

Als User Interface Framework für Java empfehlen wir **JavaFX**. Sie können aber auch andere Frameworks verwenden oder eine Konsolenapplikation entwickeln.

1. Bouncing Balls 2020 (3)

Implementieren Sie einen Pandemie Simulator der Individuen als "Bouncing Balls" darstellt. Als Vorlage kann der bekannte Artikel der Washington Post dienen:

https://www.washingtonpost.com/graphics/2020/world/corona-simulator/

Weitere Quellen und Inspiration:
https://viralballs.com/
https://observablehq.com/@a-lexwein/bouncing-balls-not-a-model-for-virus-transmission
https://www.youtube.com/watch?v=NKMHhm2Zbkw

Überlegen Sie sich Parameter zur Steuerung der Simulation, die durch User Eingriffe manipuliert und verändert werden können. Ermöglichen Sie eventuell auch das Hinzufügen von Hindernissen etc.

2. Plagiatschecker (3)

Schreiben Sie ein Programm, welches die Differenz eines Dokuments zu *n* anderen Dokumenten ermittelt. Dabei soll ein Algorithmus oder eine Heuristik entwickelt werden, der/die bestimmt in wieweit die Dokumente voneinander abweichen. Bestimmen Sie einen Grenzwert ab wann es sich vermutlich um ein Plagiat handelt.

Hinweis: Sie können git diff verwenden, um die Ähnlichkeit von zwei Text Dokumenten auszuwerten. Die Ergebnisse können dann für die Heuristik eingesetzt werden.

https://git-scm.com/docs/git-diff
https://www.atlassian.com/de/git/tutorials/saving-changes/git-diff

Das Programm soll über ein geeignetes User Interface verfügen und die Ergebnisse der Analyse übersichtlich darstellen.

3. Game of Life (5)

(maximal 2 Gruppen)

John Horton Conway hatte die Idee für diese Simulation basierend auf zellulären Automaten. Dabei wird zwischen lebenden und toten Zellen sowie Zustandsübergängen (Leben -> Tod & Tod -> Leben) unterschieden. Der Zustand zum Zeitpunkt *t+1* hängt dabei immer vom Zustand der benachbarten Zellen im Zeitpunkt *t* ab. Eine Beschreibung des Lebenszyklus / Generationenwechsels findet sich etwa hier:

https://en.wikipedia.org/wiki/Conway%27s_Game_of_Life

- Die Simulation soll Konfigurationen laden und speichern können.
- Simulationsgeschwindigkeit einstellbar / Unterstützung manueller Schritte
- Pause / Next Step / Play
- Möglichkeit zum Erstellen von Startkonfigurationen (Editor)
- Weltgröße konfigurierbar / einstellbar
- Denken Sie über mögliche Erweiterungen nach und implementieren Sie diese (5 Personen Gruppe)

4. Port Scanner (3)

(maximal 2 Gruppen)

Manchmal kann es nützlich sein, die offenen Ports eines Servers herauszufinden. Schreiben Sie ein Programm, das möglichst rasch und unauffällig die Ports eines Servers scannt. Vertiefen sich ein wenig in der Theorie der Java Klassen oder setzen sie ggf. eine externe Bibliothek ein.

Das User Interface soll die Port Range sowie die Server Adresse entgegennehmen. Fügen Sie weitere Eingabeparameter, wenn diese sinnvoll erscheinen (max Threads, Timeouts,...). Eine Fortschrittsanzeige soll die gescannten Ports wiedergeben bzw. Ausnahmen anzeigen. Überlegen Sie hier eine sinnvolle Visualisierung. (auch wie viele Threads gerade parallel im Einsatz sind. Wie viel Zeit verstrichen ist.) Welche Scan-Strategien lassen sich in Bezug auf Server umsetzen? Dokumentieren Sie diese.

Nach Abschluss des Scans kann ein Logfile geschrieben werden. Logfiles sollen auch geöffnet und übersichtlich dargestellt werden.

5. Langton's Ant (5)

(maximal 2 Gruppen)

Langton's Ant simuliert das komplexe Verhalten einer Ameise auf Basis einfacher fester Verhaltensregeln (Turing Machine). Eine Beschreibung des Verhaltens findet sich unter:

https://en.wikipedia.org/wiki/Langton%27s_ant

Implementieren Sie eine Simulationsumgebung. Die Größe der Ebene soll über Parameter einstellbar sein, oder dynamisch wachsen können (Zoom). Die Ameise lässt sich manuell auf dem Feld platzieren. Weitere einstellbare Parameter im Userinterface: Simulationsgeschwindigkeit und Anzahl der Simulationsschritte. Das Ergebnis der Simulation kann gespeichert und geladen werden.

Überlegen Sie sich Erweiterungen in Hinblick auf Farben.

6. Web Bot (3)

(maximal 2 Gruppen)

Bots (Web Bots, Chat Bots,...) übernehmen in der Regel Tätigkeiten die wir nicht manuell ausführen wollen. Implementieren Sie beispielsweise einen Bot der in regelmäßigen Zeitabständen Informationen von einer Webseite abruft und mit Zielvorgaben vergleicht. Über Alerts werden User über wichtige Änderungen informiert und diese angezeigt. Der Bot soll über ein Userinterface konfigurierbar sein. Benachrichtigungen sollen zum Beispiel erscheinen wenn ein neuer Bericht zu einem bestimmten Thema online geht oder wenn sich der Preis eines bestimmten Artikels ändert. Sie können den Anwendungsfall frei wählen. Die Tests können auch über eine eigene, einfache, kleine Webseite erfolgen.

7. JAVA KURS (3)

(maximal 3 Gruppen)

Erstellen Sie Java Kurse in IntelliJ zu zwei Themengebieten. Etwa Vererbung, Polymorphie, Interfaces, Kapselung, Exceptions etc. Der Kurs soll inkrementell aufgebaut sein und die Konzepte in mehreren Schritten aufgebaut näher bringen.

https://plugins.jetbrains.com/plugin/10081-edutools
https://www.jetbrains.com/help/education/educator-start-guide.html

Einen bestehenden Kurs selbst ausprobieren: File -> Browse Courses (stepik.org Account notwendig um an Kursen teilzunehmen)

8. ALLES IN ORDNUNG (1-3)

(maximal 2 Gruppen)

Verwaltung von Sammlungen (Bücher, CDs, DVDs, …) mit Textfiles (JSON oder YAML) und GUI (JavaFX). Das Programm soll die folgenden Möglichkeiten bieten:

- Einträge hinzufügen (z.B. Buch)
- Einträge ändern
- Einträge löschen
- Einträge suchen
- Einträge als Liste ausgeben am Bildschirm

9. TEXT EDITOR (3-5)

(maximal 2 Gruppen)

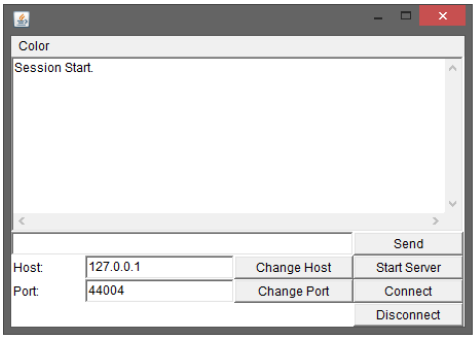
Schreiben Sie einen einfachen Texteditor. Der Editor soll zumindest folgende Funktionen umfassen:

- Text Dokument öffnen
- Dokument sichern
- Neues Dokument erstellen
- Dokument bearbeiten
- Text suchen und die Position ausgeben
- optional Schlüsselwörter farblich hervorheben

10. SIMPLE CHAT TOOL (3)

(maximal 2 Gruppen)

Der User soll in der Lage sein eine IP Adresse (oder ev. Hostnamen) und Port eingeben zu können und sich somit zu einem Gegenüber verbinden können. Sobald die beiden User verbunden sind, soll jede Nachricht, die in ein Textfeld eingegeben wird, verschickt werden. Empfangene Nachrichten sollen angezeigt werden. (ähnlich, wie bei WhatsApp, iMessage, SMS, Telegram, Slack, ...). Ein mögliches Beispiel eines einfachen Interfaces wäre:



Tutorial: https://www.javaworld.com/article/2076864/java-concurrency/building-an-internet-chat-system.html

11. DOWNLOAD MANAGER (5)

(maximal 2 Gruppen)

Dieses kleine Tool soll jeden Content eines bestimmten Typs einer Webseite herunterladen können. Content hinter Logins kann, muss aber nicht berücksichtigt werden. Es genügt einfach HTML Seiten nach Links zu parsen und auf einen bestimmten Typ zu filtern (z.B. PDF, PNG, MP4, ...). Das Tool soll alle Dateien anzeigen, die es von der eingegebenen Webseite herunterladen wird. Man soll dann die Möglichkeit haben einen Ordner auszuwählen, in dem dann alle Dateien abgespeichert werden.

12. WEB CRAWLER (3-5)

(maximal 2 Gruppen)

Ähnlich wie der Download Manager soll auch der WebCrawler Webseiten besuchen. Allerdings soll er alle Links abspeichern, die in der Seite gefunden werden. Diese sollen als JSON oder YAML Datei serialisiert werden. Es soll auch möglich sein die ersten x Links einer Google Suche herauszufiltern.

13. ELEVATOR SIMULATOR (5)

(maximal 2 Gruppen)

Wieviele Aufzüge benötigt ein Hotel mit 20 Stockwerken und 20 Doppelzimmern pro Etage bei 80% Auslastung damit die Gäste nicht länger als x Minuten am Fahrstuhl warten? Was passiert wenn 50% der Gäste zwischen 8 und 10 Uhr Vormittags in den 1. Stock zum Frühstück wollen? Was passiert wenn Aufzüge Personen nicht einsammeln? Diese oder ähnliche Fragestellungen sollen sich mit Hilfe des Aufzugssimulators beantwortet werden. Treffen Sie nötige Annahmen oder Vereinfachungen, wie z.B. die Liftgeschwindigkeit.

Das Programm soll es erlauben, dass Parameter geändert werden können (Stockwerke, max. Wartezeit, Zimmer pro Etage, Auslastung, ...) und dann die Ergebnisse aktualiseren (Anzahl Aufzüge, durchschnittliche Wartezeit, ...)

14. DIE ANALOGE UHR (1)

(maximal 2 Gruppen)

Implementieren Sie eine analoge Uhr mit Sekunden, Minuten und Stundenzeiger. Verpassen Sie ihrer Uhr nach Möglichkeit unterschiedliches Aussehen.

15. EIERUHR (1-3)

(maximal 2 Gruppen)

Beim Kochen ist es oft erforderlich mehrere Timer zu haben. Etwa für Brötchen aufbacken, Eier weich kochen etc. Entwickeln Sie eine elektronische Eieruhr die zeitgleich mehrere Timer verwalten und starten kann. Den Timern sollen unterschiedliche Warnsignale zugeordnet werden können. Im Idealfall passen die Warnsignale automatisch zu der gewählten Aufgaben (z.B. Brötchen aufbacken) ☺
Die Anzeige könnte auch graphisch erfolgen: Zum Beispiel eine animierte Sanduhr.

16. WETTER APPLIKATION (3)

(maximal 2 Gruppen)

Das Tool soll es ermöglichen, für einen gewählten Ort das aktuelle Wetter und das Wetter für die nächsten Tage anzuzeigen. Dazu soll das Wetter von einem Server geladen werden. Optional kann das Tool auch versuchen den aktuellen Ort selbst zu bestimmen. Wenn möglich kann das Wetter auch graphisch dargestellt werden.
Wetter Server Beispiele:

- https://openweathermap.org/api
- https://developer.yahoo.com/weather/
- http://wetter-api.com/

17. BULK RENAMER (1)

(maximal 2 Gruppen)

Das Tool soll es erlauben einen Ordner auszuwählen und ein Namensfilter für Dateien anzugeben (z.B. .pdf oder eventuell eine Regular-Expression). Danach soll das Tool alle Dateien, die dem Filter in diesem Ordner entsprechen umbenennen. Dazu soll der User ein Prefix eingeben können (z.B. „Bild_") an welches eine fortlaufende Nummer und die Dateiendung angehängt wird.

18. CONFIGURABLE CASH REGISTER (5)

(maximal 2 Gruppen)

Die elektronische Registrierkassa ist in aller Munde. Die Konfiguration und Anordnung von Buttons und Bedienelementen aber nicht immer an die Bedürfnisse des Geschäftsmodells angepasst. Entwickeln Sie eine Konfigurationsdatei (JSON oder YAML) die Anordnung, Größe, Bezeichnung, Typ (etwa Produkt oder Operation) der Bedienelemente beinhaltet und die Kassa danach entsprechend darstellt. Die Beispielimplementierung soll einfache Produkte und einen Rechner beinhalten und sich etwa für ein Restaurant eignen.

FH CAMPUS WIEN
UNIVERSITY OF APPLIED SCIENCES

VERSPIELT

Im folgenden sind beispielhaft einige Spiele für die Projektarbeit genannt. Nach Möglichkeit eine bis maximal 2 Gruppen pro Spiel.

Alle Spiele können mit JavaFX umgesetzt werden.

- Wenn möglich sollten die Spiele über ein Menü zur Steuerung des Spiels verfügen (Anhalten, Neustart, etc...).
- Animationen sind dort wo es Sinn macht und der BenutzerInnenfreundlichkeit dienen einzusetzen. Bei einigen Spielen erfordert die Spielmechanik von Haus aus Animationen.
- Computerspiele eignen sich vor allem für größere Gruppen, da diese in der Regel einen höheren Arbeitsaufwand bedeuten (Animation, Timer, etc.).

19. SNAKE (5)

You know Snake ;)

20. VIER GEWINNT (5)

Eine Spielbeschreibung befindet sich unter:
https://de.wikipedia.org/wiki/Vier_gewinnt

Das Spiel soll das Spielfeld anzeigen und der User soll gegen den Computer (oder einen anderen User) spielen können.

21. BREAKOUT (5)

Eine Spielbeschreibung für den Klassiker der Arcade Spiele findet sich unter:
https://de.wikipedia.org/wiki/Breakout_(Computerspiel) und
https://cs.stanford.edu/people/eroberts/courses/cs106a/handouts/25-assignment-3.pdf

22. BLACKJACK (3)

Eine Spielbeschreibung befindet sich unter:
https://de.wikipedia.org/wiki/Black_Jack

Das Programm soll die Karten des Users anzeigen und er/sie soll gegen den Computer spielen können.

23. BATTLE SHIP (3)

Eine Spielbeschreibung befindet sich unter:
https://de.wikipedia.org/wiki/Schiffe_versenken

Hier sollten folgende Spielmodi umgesetzt werden: Mensch vs. Mensch und Mensch vs. Computer.
Das Programm soll das Spielfeld (Kästchen) anzeigen und entsprechend markieren.

24. HANGMAN (3)

Eine Spielbeschreibung befindet sich unter:
https://de.wikipedia.org/wiki/Galgenm%C3%A4nnchen

Der Computer wählt ein Wort aus einer Liste von Nomen, die entweder aus einem File (JSON, YAML,...) eingelesen oder über ein Service wie https://www.randomlists.com/, erstellt werden. Es könnte auch aus einer Kategorie gewählt werden.

Das Programm zeigt dann die aufgelösten Buchstaben und die noch möglichen Versuche an (vorzugsweise graphisch)

25. CHESS (5)

Implementieren Sie Schach für zwei menschliche Spieler. Es soll das Spielbrett angezeigt werden und nur gültige Züge zugelassen werden.

26. CHECKER (3)

Oder auch besser bekannt als Dame. Siehe auch: https://de.wikipedia.org/wiki/Dame_(Spiel).
Hier sind ebenfalls nur zwei menschliche Spieler zu unterstützen. Keine KI. Es soll das Spielbrett angezeigt werden und nur gültige Züge zugelassen werden.

27. QUIZ MAKER (3)

Wer wird Mister Java? Implementieren Sie ein Java Quiz im Stil von „Wer wird Millionär". Fragen und mögliche Antworten sind dabei mittels JSON oder YAML abzuspeichern.

28. MINESWEEPER (5)

Logik und Ablauf des Spiels ist auf https://de.wikipedia.org/wiki/Minesweeper genau beschrieben. Das Programm soll das Spielfeld mit den aufgedeckten Feldern und den Minen anzeigen.

29. MEMORY PUZZLE (3)

Wer kennt nicht Memory aus seiner Kindheit? ☺ Diese Spiel lebt von eigenwilligen Bildern und Symbolen. Stellen Sie die Puzzle Teile (Bilder) im Programm dar und erlauben Sie das Umdrehen von jeweils zwei Teilen. Es sollten zwei User gegeneinander spielen können.