C++ Sprache

1: Simulationsumgebung für Würfelspiele (C++)

Ziel

Ein Framework für verschiedene Würfelspiele (z. B. Kniffel/Yatzy, Mensch ärgere dich nicht, Risiko-Würfe), mit Statistikmodul.

Anforderungen

- Modellierung von Würfeln und Regeln
- Spieler- und Rundenlogik
- Speichern von Partien und Statistiken (z. B. Siegquote, Würfelverteilungen)
- Optional: Spielmodus gegen KI oder grafische Ausgabe

Projektaufgabe: Konsolenbasierte Schach-Engine in C++

Einleitung / Zielsetzung

In diesem Projekt implementieren Sie ein **konsolenbasiertes Schachspiel** mit einer Spiellogik, Benutzersteuerung und einer einfachen **Computer-KI**. Ziel ist es, die Spielregeln korrekt umzusetzen und über das Terminal ein vollständiges Match gegen einen Spieler oder eine einfache Engine zu ermöglichen.

Fachliche Anforderungen

Funktionale Anforderungen (mindestens zwei Anforderungen müssen abgedeckt werden)

1. **Spiellogik**
   - Darstellung des Schachbretts mit ASCII- oder Unicode-Zeichen
   - Regelkonforme Zuggenerierung (inkl. Rochade, En Passant, Umwandlung)
   - Erkennung von Schach, Matt, Patt
2. **Spielmodi**
   - Spieler gegen Spieler
   - Spieler gegen KI (einfache Bewertung mit Materialwert)
3. **Speicherung**
   - Optional: Laden und Speichern von Spielständen
   - Log der Züge (PGN oder einfache Notation)
4. **KI (Minimale Stufe)**
   - Minimax mit fixer Tiefe (z. B. 2)
   - Bewertung nach Material (Bauer = 1, Dame = 9 usw.)
5. **Optional (Bonus)**
   - Rückgängig-Funktion
   - Zufallszüge bei gleicher Bewertung

2: Lernkarten-Tool mit Spaced-Repetition (C++)

Ziel

Ein CLI-Tool zur Verwaltung und Abfrage von Lernkarten mit Zeitintervall-Steuerung (wie Anki, nur einfach).

Anforderungen (mindestens zwei Anforderungen müssen abgedeckt werden)

- Verwaltung von Karten (Frage, Antwort, Kategorie)
- Statistik: Wiederholungsrate, Bewertung (leicht/schwer)
- Algorithmus für Wiederholungsintervall (z. B. SM-2)
- Optional: Import/Export, Tagging, CSV-Unterstützung

3: Rennspiel-Simulator (Text- oder ASCII-Grafik, C++)

Ziel

Ein rundenbasiertes Auto-/Kartsystem mit Fahrphysik, Strecken-Layout und einfacher KI.

Anforderungen (mindestens zwei Anforderungen müssen abgedeckt werden)

- Strecke als Raster (ASCII)
- Fahrzeuge mit Geschwindigkeits-/Richtungswerten
- Kollisionserkennung, Rundenzeit, einfache KI-Gegner
- Optional: Power-Ups, Mehrspielermodus (nacheinander), Replay-Funktion

---

4: Textbasiertes Spiel (C++)

Ziel

Ein interaktives, konsolenbasiertes Rollenspiel mit Kämpfen, Inventar, Levelsystem und zufälligen Events.

Anforderungen (mindestens zwei Anforderungen müssen abgedeckt werden)

- Spielwelt mit mehreren Räumen / Zonen
- Held mit HP, XP, Angriff/Werte
- Gegner mit KI (Zufallsverhalten oder Regeln)
- Inventarsystem (Gegenstände, Heiltränke)
- Speicher-/Ladefunktion
- Optional: ASCII-Grafik, Minimap, Quests

5: Netzwerk-Monitoring-Tool (C/C++)

Ziel

Ein Tool zur Überwachung und Analyse von Netzwerkverkehr auf einem lokalen Rechner (ähnlich Wireshark Light).

Anforderungen (mindestens zwei Anforderungen müssen abgedeckt werden)

- Erfassen von ein- und ausgehenden Paketen (z. B. mit `libpcap`)
- Anzeige von IP-Adressen, Ports, Protokollen
- Filterung nach Protokoll (TCP, UDP, ICMP)
- Statistiken (Traffic pro Sekunde, Verbindungen)
- Optional: Export als CSV, grafische Auswertung mit externem Tool

6: Simulationsumgebung für Würfelspiele (C++)

Ziel

Ein Framework für verschiedene Würfelspiele (z. B. Kniffel/Yatzy, Mensch ärgere dich nicht, Risiko-Würfe), mit Statistikmodul.

Anforderungen (mindestens zwei Anforderungen müssen abgedeckt werden)

- Modellierung von Würfeln und Regeln
- Spieler- und Rundenlogik
- Speichern von Partien und Statistiken (z. B. Siegquote, Würfelverteilungen)
- Optional: Spielmodus gegen KI oder grafische Ausgabe