Coding for Fun

# Mögliche Projekte

1. **Minesweeper**

Es soll das bekannte und beliebte «Bombensuchspiel» nachprogrammiert werden: Auf einem Spielbrett werden verdeckt Bomben versteckt. Der Spieler muss alle Felder aufdecken, ohne auf eine Bombe zustossen. Zahlen geben an, wie viele Bomben in unmittelbarer Nähe zu finden sind.



* Zufallszahlen
* Enum Direction
* Visualisierung als ASCII-Grafik
* Spielfeld
  + Generieren basierend auf Bombenfeldern und Ausrechnen Nachbarschaft
  + Aufdecken von mehreren Feldern => Rekursion

1. **Tic Tac Toe**

Es soll das bekannte und beliebte Spiel nachprogrammiert werden: Dabei wechseln sich zwei Spieler symbolisiert durch O und X ab und versuchen als erster drei gleiche Symbole in einer Reihe zu erzielen.

**Ein Bild, das Text, Shoji, Kreuzworträtsel, Uhr enthält.

Automatisch generierte Beschreibung**

* Visualisierung als ASCII-Grafik
* Abwechselnder Spielablauf mit Test auf Sieg und Unentschieden

1. **Worträtsel**

In dieser Aufgabe geht es um das Generieren von Worträtseln, die vermutlich aus Zeitschriften bekannt sind: Die Herausforderung besteht darin, in einem wilden Mix aus scheinbar zufälligen Buchstaben verschiedene dort versteckte Begriffe zu finden. Unsere Aufgabe ist es nun, ein solches Worträtsel basierend auf einer Liste von Wörtern zu generieren. Es soll etwa wie nachfolgend gezeigt aussehen:

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

* Datei einlesen / DB-Handling
* Enum Direction
* Zufallszahlen
* Flexibilität 2-D-Array
* Visualisierung
* Algorithmik

1. **Highscore-Verwaltung**

Stellen wir uns eine x-beliebige Spieleapplikation aus den obigen vor, die es einem Spieler erlaubt, entsprechende Punktzahl vorausgesetzt, sich in einer Highscore-Liste zu verewigen.

Nehmen wir weiterhin an, die Spielstände wären kommasepariert etwa wie folgt in einer Datei *highscores.csv* gespeichert. In diesem Beispiel sind bewusst auch fehlerhafte Einträge dargestellt, die dazu dienen, die Implementierung einer robusten Fehlerbehandlung zu demonstrieren und zu testen:

**# Name, Punkte, Level**

**Matze, 1000, 7**

**Peter, 985, 6 ÄÖÜßöäü, 777, 5**

**# Fehlender Level**

**Peter, 985,**

**# Falsches Format des Levels**

**Peter, 985, A6**

* **Spring Boot App (Spring Initializr)**
* **Highscores aus csv einlesen**
* **Fehlerbehandlung**
* **REST-API mit CRUD**
* **Optional: JPA mit HSQL/H2 DB**

PARKHAUS

**Sokoban**

Es soll das bekannte und beliebte «Kästchenschiebspiel» nachprogrammiert werden: Auf einem Spielbrett müssen Kisten auf Zielfelder verschoben werden. Diamanten kann man nur verschieben, wenn keine Wand oder kein Diamant im Weg ist.

* Level einlesen
* Spielfeld- Modellierung => Wiki

#############

## #DD <#

#### #D# OO #

# <# #D O O ## P#

#B # # #

##### ###############

# -- Wand

P – Player

D – Ziel

O – Diamond

* Enum Direction
* Visualisierung

**2-D Racer**

<https://copyassignment.com/simple-car-race-game-in-java/>

<https://www.youtube.com/watch?v=6td_zpQC1U4>

<https://www.sourcecodester.com/javascript/15081/25d-road-racing-game-using-javascript-free-source-code.html>