

## 2. Übungszettel in Software Engineering

### Teilaufgabe 1

<b>Name</b>	Spielfeld variabler Größe
<b>Status</b>	Bereits mit der letzten Abgabe erledigt.

### Teilaufgabe 2

<b>Name</b>	Variable Spielerzahl
<b>Status</b>	Bereits mit der letzten Abgabe erledigt.

### Teilaufgabe 3

<b>Name</b>	Menü um KI-Einstellung erweitern
<b>Motivation</b>	Die Einstellungsmöglichkeit ist Bedingung dafür, dass der Spieler mit/gegen KIs spielen kann.
<b>Funktion</b>	Beim Start des Spieles möchte ich als Spieler einstellen können, ob ich gegen natürliche Spieler und/oder gegen/mit Unterstützung einer KI spielen möchte.
<b>Akzeptanzkriterien</b>	Die möglichen Einstellungen muss der einstellende Spieler intuitiv verstehen.
<b>Schätzung</b>	1,5 Stunden
<b>Begründung</b>	Wir wollen den Code in eine zusätzliche Klasse auslagern und vermuten Komplikationen bei der Integration in die bestehenden Abläufe.
<b>Tatsächlicher Aufwand</b>	Als problematisch erwiesen sich die vielen verschiedenen Spielmodi (Spieler mit/ohne Assistenz gegen KI Random oder Minmax)
<b>Reflektion</b>	2

### Teilaufgabe 4

<b>Name</b>	Abstrakte KI-Klasse erstellen
<b>Motivation</b>	Es wird eine Schnittstelle für KIs mit verschiedenen Strategien benötigt. Außerdem soll Redundanz vermieden werden.
<b>Funktion</b>	Im Spielablauf sollen - unabhängig davon welche KI gerade am Zug ist - feste Methoden aufzurufen sein.
<b>Akzeptanzkriterien</b>	Auch wenn es weitere KIs geben soll, sollten sich an der eigentlichen Spiel-Engine keine Änderungen mehr ergeben.
<b>Schätzung</b>	1 Stunde
<b>Begründung</b>	Es muss geprüft werden, welche Methoden die Abstrakte Klasse zur Verfügung stellen soll.
<b>Tatsächlicher Aufwand</b>	1
<b>Reflektion</b>	Keine unerwarteten Probleme

## Teilaufgabe 5

<b>Name</b>	KIRandom Klasse erstellen (zufälliger Zug)
<b>Motivation</b>	Es soll eine KI geringer Schwierigkeitsstufe geben, die zufällig eine Mauer wählt.
<b>Funktion</b>	Ich will als Spieler gegen eine leichte KI spielen.
<b>Akzeptanzkriterien</b>	Die Züge der KI sollen zufällig wirken.
<b>Schätzung</b>	1 Stunde
<b>Begründung</b>	Nicht besonders kompliziert.
<b>Tatsächlicher Aufwand</b>	1
<b>Reflektion</b>	Keine unerwarteten Probleme

## Teilaufgabe 6

<b>Name</b>	KIMinMax Klasse erstellen (Zug nach Min-Max-Algorithmus)
<b>Motivation</b>	Es soll eine KI mittlerer Schwierigkeit geben, die Züge nach dem Minimax-Algorithmus wählt.
<b>Funktion</b>	Ich will als Spieler gegen eine KI mittlerer Schwierigkeit spielen, die ihre Züge so wählt, dass für sie das optimale Ergebnis herauskommt, ohne meine Züge zu berücksichtigen.
<b>Akzeptanzkriterien</b>	Der Spieler soll eine gewisse Intelligenz hinter den Zügen erkennen können.
<b>Schätzung</b>	4 Stunden
<b>Begründung</b>	Zunächst wird Einarbeitungszeit in den Algorithmus benötigt. Außerdem muss die gesamte Historie durchgespielt und gespeichert werden.
<b>Tatsächlicher Aufwand</b>	8 Stunden
<b>Reflektion</b>	Die Einarbeitungszeit in den Algorithmus war zwar kürzer als erwartet. Dafür fiel uns unsere bisherige Implementierung der Karte auf die Füße. Die Auslagerung in eine Extraklasse, die alle benötigten Methoden implementiert kostete viel Zeit. Wir hoffen, dass die Neuimplementierung es uns erleichtert, zukünftige Anforderungen umzusetzen.