Testplan 4Way4

Testplan von Lorenz Schwab und Lukas Koddenberg

1 Testgegenstand

Testgegenstand ist das in der SE1-Vorlesung vorgestellte Spiel „4 Way 4“, welches von jedem Team programmiert werden soll. Das Spiel kann als eine Variante des beliebten Brettspielklassikers „vier gewinnt“ angesehen werden.

2 Umfang

Zu testen sind folgende funktionale und nicht funktionale Anforderungen:

- Spielfeldgröße von $7 * 7$ bis $10 * 10$. Es muss nicht quadratisch sein.
- jeder Spielstein (Token) belegt genau ein Feld.
- Spielmodus Spieler 1 gegen Spieler 2
- Die Spielregeln werden eingehalten.
- Spielmodus Spieler 1 gegen CPU:
  CPU in drei verschiedenen Schwierigkeitsstufen

3 Spielregeln:

1. Wird ein Token eingeworfen, so werden alle auf dem Feld befindliche Tokens so weit in die Richtung gerückt, in die der Token das Feld betreten hat, wie möglich. Die Spielfeldränder sind dabei die Höchstgrenze.
2. Gewonnen hat der Spieler, der als erstes vier Tokens in einer Reihe hat. Eine Reihe kann horizontal, vertikal und diagonal sein.
3. Das Spiel endet, wenn ein Spieler oder die CPU gewinnt, oder wenn alle Felder des Spielfeldes belegt sind. Dann gewinnt der Spieler, der den letzten Zug machen konnte.
4. Spielmodus Spieler gegen CPU:
   Der Spieler entscheidet, wer beginnt.

4 Testen der funktionalen Anforderungen

- Spielfeldgröße von $7 * 7$ bis $10 * 10$. Es muss nicht quadratisch sein.
- jeder Spielstein (Token) belegt genau ein Feld.
- Spielmodus Spieler gegen Spieler
- Die Spielregeln werden eingehalten.

4.1 Testen der funktionalen Anforderungen

4.1.1 Testen auf dem Papier (Black - Box - Test)

Die ersten Tests wurden von den Teammitgliedern mit Stift und Papier
(meist auf einem $7 * 7$ - Spielfeld) durchgeführt, um die zugrundeliegende Funktionsweise des Spiels zu verstehen. Außerdem wurde eine Runde mit dem Kunden zusammen an der Tafel gespielt, um Verständnisfragen zu klären.

4.1.2 JUNIT

Die meisten Tests sind aus Gründen der Automatisierung mit JUNIT geschrieben, denn bei über 90% der Fälle macht es einfach keinen Sinn, manuell zu testen.
Vor allem müssen die Extremfälle getestet werden. Extremfälle sind zum Beispiel das Abfangen von Exceptions bei Falscheingaben, um zu verhindern, dass das Spiel aufgrund einer Falscheingabe abstürzt.

4.2 Tests:

4.2.1 Testkriterien

Die Spielfeldgröße muss zwischen $7 * 7$ und $10 * 10$ liegen.
Das Spielfeld kann auch rechteckig sein, nicht nur quadratisch.
Jeder Token belegt ein Feld.
Spielmodus Spieler 1 gegen Spieler 2.
Die Spielregeln werden eingehalten.
Integritätstest: Es werden zwei KIs instantiiert. Diese müssen dann gegeneinander spielen. Eventuell auftretende Fehler müssen behoben werden, bis beide KIs problemlos gegeneinander spielen können.

4.2.2 Testende

Die Spielfeldgröße kann im Bereich $7 * 7$ bis $10 * 10$ ganzzahlig festgelegt werden.
Das Spielfeld kann rechteckig sein, nicht nur quadratisch.
Jeder Token belegt ein Feld.
Es gibt den Spielmodus Spieler 1 gegen Spieler 2.
Die Spielregeln werden eingehalten.
Die KIs können problemlos instantiiert werden und spielen gegeneinander. Eine der KIs gewinnt das Spiel. Beide KIs erkennen das Ende des Spiels.

4.2.3 Testabbruch

Die Spielfeldgröße liegt nicht zwischen $7 * 7$ und $10 * 10$ oder ist nicht ganzzahlig.
Das Spielfeld kann nur quadratisch sein oder andere Formen als ein Rechteck annehmen.
Jeder Token belegt mehr oder weniger als ein Feld.
Spielmodus Spieler 1 gegen Spieler 2 existiert nicht.
Die Spielregeln werden nicht eingehalten.
Es treten Fehler auf, die das Spiel sofort beenden. Eine KI denkt, das Ende des Spiels ist erreicht, aber die andere KI spielt weiter. Sonstige Fehler treten auf und das gewünschte Ergebnis wird nicht erzielt.

4.3 Weitere Tests:

Ferner müssen noch zu jeder Methode in jeder Klasse Tests erstellt werden, die die ordnungsgemäße Funktion der Methoden sicherstellen. Dies wird folgendermaßen geschehen:

1. Instantiieren der nötigen Objekte
2. Manipulation der jeweiligen Attribute der Objekte

   ggf. Herbeiführen von Ausnahmen und behandeln dieser
3. Abfragen von Zuständen, ausgelösten Ausnahmen etc.
4. Löschen der Objekte

Die obigen Schritte lassen sich für mehrere Variablen durchführen; dadurch müssen auch mehrere dieser doch sehr ähnlichen Tests geschrieben werden. Das Prinzip jedoch ändert sich nicht sonderlich. So ergeben sich dann mehrere Testfälle für eine Methode.
Durch die Tests wird versucht, eine möglichst hohe Testabdeckung zu erlangen, wenngleich eine 100% - ige Testabdeckung nahezu unmöglich ist. Das Ziel von Team Jan ist es, eine Testabdeckung von $\geq 80\%$ zu erreichen.
In diesem Dokument werden die Testmethoden, die sich besonders von den obigen Standardtests unterscheiden, aufgeführt:

4.3.1 Besondere Tests:

noch keine besonderen Tests eingefügt