Projektkonzept

9 Holes

Gabriel Schafflüztel

Inhalt

[Grundidee 3](#_Toc529470790)

[Klassen 4](#_Toc529470791)

[Iterationen 5](#_Toc529470792)

[Beispiel eines Spiels 6](#_Toc529470793)

[Daten 8](#_Toc529470794)

# Grundidee

Die Aufgabe besteht darin, ein Java Projekt gemäss Vorgaben der WISS umzusetzen. Ich entschied mich für ein Spiel namens 9 holes. Die Spielregeln funktionieren folgendermassen:

Material

Es findet sich ein Brett vor, welches 9 Löcher besitzt. Es benötigt zwei Spieler, wobei jeder 3 Spielsteine besitzt.

Platzierungsphase

Je ein Spieler platziert rundenbasiert eines seiner Spielsteine in ein Loch, bis keine Spielsteine übrig sind.

Bewegungsphase

Je ein Spieler bewegt rundenbasiert seine Steine so, dass schlussendlich 3 Steine des Spielers nebeneinanderliegen. Die Spieler dürfen nicht auf Felder, die von gegnerischen Steinen besetzt sind und gewinnen kann man nur dann, wenn die eigenen Steine vertikal oder horizontal 3 Löcher besetzen.

# Klassen

Grundgerüst

Zuerst werde ich das Projekt in Eclipse aufbauen. Mein zukünftiges Projekt sollte aus einer Klasse Board, wo alle Spielfelddaten gespeichert werden, zwei Player Klassen, eine für den menschlicheren Spieler und eine für den KI, eine Display Klasse und einem Controller, der alles koordiniert. Die Klassen gesamt werden womöglich ungefähr so aussehen:

Player

numPieces : Integer

nextDrop()

nextMove()

2

Human

AI

Controller

Board

pieces[] : Integer

fields[] : Integer

setPiece()

movePiece()

IsFieldEmpty()

Display

1 1

1

1…\*

1

Vielleicht füge ich eine Klasse namens Piece hinzu, um Spielsteine im Board zu initialisieren, jedoch bin ich mir darüber noch nicht im Klaren.

# Iterationen

Ich werde womöglich nachfolgende Iterationen durchführen:

Projektstart

In dieser Iteration werde ich das Projekt starten, das Verzeichnis erstellen und alle Dateien erzeugen.

Spielbrett

Hier werde ich die Klasse Board erstellen und die Klasse Display soweit, bis ich das Spielbrett auf der Konsole ausgeben kann.

Spieler

An diesem Zeitpunkt werde ich die abstrakte Klasse Player erstellen.

Mensch

Anschliessend versuche ich, die Klasse Human zu kreieren.

Human vs. Human

Hier werde ich den Programmcode so erweitern, dass ich gegen mich selbst spielen kann, sodass ich die generelle Funktionalität des Spieles proben kann.

AI

Schlussendlich programmiere ich den AI. Dies bedeutet, dass ich als erstes ein AI erstelle, der seine Bewegungen zufällig ausführt. Erst danach erstelle ich ein funktionsfähiger AI, der mögliche Bewegungen mathematisch auf einer Skala bewertet und anschliessend auf schlagbare Art und Weise fair spielt.

Tests

Um das Programm zu testen, werde ich alle relevanten Variabeln in JUnit beifügen und ihre Korrektheit überprüfen.

# Beispiel eines Spiels

Unten füge ich ein möglicher Durchlauf, jedoch dient dieser nur zur Veranschaulichung und um auf Elemente aufmerksam zu machen, die zu programmieren sind.

\_\_\_ \_ \_ \_\_\_\_ \_ \_\_\_\_\_\_ \_\_\_\_\_

/ \_ \ | | | | / \_\_ \ | | | \_\_\_\_| / \_\_\_\_|

| (\_) | | |\_\_| | | | | | | | | |\_\_ | (\_\_\_

\\_\_, | | \_\_ | | | | | | | | \_\_| \\_\_\_ \

/ / | | | | | |\_\_| | | |\_\_\_\_ | |\_\_\_\_ \_\_\_\_) |

/\_/ |\_| |\_| \\_\_\_\_/ |\_\_\_\_\_\_| |\_\_\_\_\_\_| |\_\_\_\_\_/

1 2 3

a( )-( )-( )

| | |

b( )-( )-( )

| | |

c( )-( )-( )

>>> You have 3 pieces left.

>>> Place a piece. e.g.(b3): b3

1 2 3

a( )-( )-( )

| | |

b( )-( )-(O)

| | |

c( )-( )-( )

>>> The computer has 3 pieces left.

>>> Computer is placing his piece.

1 2 3

a( )-( )-( )

| | |

b( )-(X)-(O)

| | |

c( )-( )-( )

>>> You have 2 pieces left.

>>> Place a piece. e.g.(b3): b3

>>> SpotError: Spot is not available.

>>> You have 2 pieces left.

>>> Place a piece. e.g.(b3): a3

1 2 3

a( )-( )-(O)

| | |

b( )-(X)-(O)

| | |

c( )-( )-( )

>>> The computer has 2 pieces left.

>>> Computer is placing his piece.

1 2 3

a( )-( )-(O)

| | |

b( )-(X)-(O)

| | |

c( )-( )-(X)

>>> You have 1 piece left.

>>> Place a piece. e.g.(b3): a2

1 2 3

a( )-(O)-(O)

| | |

b( )-(X)-(O)

| | |

c( )-( )-(X)

>>> The computer has 1 piece left.

>>> Computer is placing his piece.

1 2 3

a(X)-(O)-(O)

| | |

b( )-(X)-(O)

| | |

c( )-( )-(X)

>>> You are now finished placing pieces. Now it's time to move.

>>> MoveError: No valid move results in turn skip.

>>> Computer is moving.

1 2 3

a(X)-(O)-(O)

| | |

b(X)-( )-(O)

| | |

c( )-( )-(X)

>>> Choose a piece. e.g. (b3): b3

>>> Choose a direction. e.g. (u): u

>>> SpotError: Spot is not available.

>>> Choose a piece. e.g. (b3): b3

>>> Choose a direction. e.g. (u): l

1 2 3

a(X)-(O)-(O)

| | |

b(X)-(O)-( )

| | |

c( )-( )-(X)

>>> Computer is moving.

1 2 3

a(X)-(O)-(O)

| | |

b(X)-(O)-( )

| | |

c( )-(X)-( )

>>> Choose a piece. e.g. (b3): b3

>>> PieceError: Piece does not exist.

>>> Choose a piece. e.g. (b3): a3

>>> Choose a direction. e.g. (u): d

1 2 3

a(X)-(O)-( )

| | |

b(X)-(O)-(O)

| | |

c( )-(X)-( )

>>> Computer is moving.

1 2 3

a(X)-(O)-(O)

| | |

b(X)-(O)-( )

| | |

c(X)-( )-( )

>>> You lose.

\_\_\_ \_ \_ \_\_\_\_ \_ \_\_\_\_\_\_ \_\_\_\_\_

/ \_ \ | | | | / \_\_ \ | | | \_\_\_\_| / \_\_\_\_|

| (\_) | | |\_\_| | | | | | | | | |\_\_ | (\_\_\_

\\_\_, | | \_\_ | | | | | | | | \_\_| \\_\_\_ \

/ / | | | | | |\_\_| | | |\_\_\_\_ | |\_\_\_\_ \_\_\_\_) |

/\_/ |\_| |\_| \\_\_\_\_/ |\_\_\_\_\_\_| |\_\_\_\_\_\_| |\_\_\_\_\_/

1 2 3

a( )-( )-( )

| | |

b( )-( )-( )

| | |

c( )-( )-( )

>>> You have 3 pieces left.

>>> Place a piece. e.g.(b3):

# Daten

* Der Projektstart und die Festlegung der Projektidee ist am 7. November 2018.
* Die Abgabe des Projektkonzepts erfolgt am 9. November 2018.
* Der Dozent gibt das Konzept am 14. November 2018 frei.
* Die fertige Arbeit ist am 21. November 2018 um 16:30 abzugeben.