Programm Aufbau

Unser Programm setzt sich zusammen aus den vier Klassen Game, Board, Player und Bot.

Die **Klasse Player** verwaltet lediglich ihre Information, wie den Namen des Spielers und seine Nummer und weist sonst nur die Funktionen auf, die der Benutzer benötigt, seinen Zug auf dem Spielfeld zu machen.

Die **Klasse Bot**, erbt die Fähigkeiten vom Spieler und erhält noch die Möglichkeit dazu in 3 verschiedenen Komplexitätsgraden auf das Spiel zu reagieren.

Die Klasse Board verwaltet das Spielfeld, bietet die Möglichkeit dieses auf der Konsole zu visualisieren und kontrolliert, wenn es in Benutzung ist, seinen Zustand. Es kann erkennen welcher Spieler gewonnen hat oder ob es zu einem Unentschieden gekommen ist.   
Zudem ist das Board dynamisch anpassbar und lässt sich in Verschiedenen Größen erzeugen und kann auf verschiedene

KI-Konzept:

Es wurden 2 Bots erstellt, die die gleichen Methoden in unterschiedlicher Reihenfolge nutzen. Dadurch ist zum einen ein defensiv ausgerichteter Bot entstanden, da er zuerst schaut, ob er verteidigen kann und danach angreift. Zum anderen ist ein offensiv ausgerichteter Bot entstanden, da dieser zuerst angreift und dann verteidigt.

Zukünftig abwägen möglich

Die Bots nutzen 3 Methoden. Die erste Methode ist das Verteidigen (check\_in\_danger). Diese schaut nacheinander, ob der Gegner horizontal, vertikal oder diagonal eine Reihe hat die mindestens zwei Elemente lang ist. Wenn das der Fall ist, so berechnet sie für den spezifischen Fall ein Feld, welches den Gegner blockiert. Damit verteidigt sich der Bot frühzeitig gegen seinen Gegner.

Die zweite Methode ist das Angreifen (check\_finish). Diese ähnelt dem Verteidigen sehr, da sie die gleichen Methoden nutzt. Jedoch wird in der Methode statt wie beim Verteidigen die Nummer des Gegners hier die eigene Nummer übergeben. Die Methode schaut nun, ob der Bot selbst horizontal, vertikal oder diagonal eine Reihe hat, die mindestens zwei Elemente lang ist. Trifft dieser Fall ein so berechnet sie ein Feld, welches die Reihe verlängert und den Bot somit näher an den Sieg heranführt.

Die letzte Methode ist zum Start eines Angriffs des Bots gedacht (start\_attack). Sie schaut nur nach einem eigenen Feld und prüft dann, ob die Felder daneben frei sind. So macht der Bot in dieses seinen Zug und kann damit in der nächsten Runde einen Angriff starten.

Der defensive Bot schaut zuerst, ob er verteidigen kann (check\_in\_danger) und greift danach erst an (check\_finish). Der offensive Bot schaut zuerst ob das mmittlere Feld frei ist, danach ob er angreifen kann (check\_finish), dann ob er verteidigen muss (check\_in\_danger) und wenn er dies auch nicht muss, so startet er einen Angriff (start\_attack).