Damit der KIs später automatisiert lernen können wird eine weiter KI benötigt, die bereits spielfähig ist. Da eine der Vier-Gewinnt-Implementationen bereits eine gut KI enthielt, diente diese als Vorlage für die *NormalKI2*. [vgl. Quelle]

Der enthaltene Algorithmus basiert auf einfachen if-then-Abfragen und Schleifen. Dabei wird zunächst geprüft, ob das Spielfeld leer ist. In diesem Fall wirf sie in die mittlere Spalte um eine möglichst gute Ausgangslage zu schaffen. Wenn bereits Steine auf dem Spielfeld liegen, prüft die KI als erstes ob sie selbst gewinnen kann. Falls nicht wird abgefragt ob der Gegner gewinnen kann und der eigene Stein entsprechend so geworfen, um dies zu verhindern.

Sofern keine dieser Fälle vorliegt, wird eine defensive Spielweise gewählt. Für ein klassisches Vier-Gewinnt-Spiel wird nun geprüft, welche Züge es dem Gegner erlauben würden drei bzw. zwei seine Steine nebeneinander zu platzieren. Alle entsprechenden Züge werden als Spalten in einem Array *possibleSolutions* zwischengespeichert. Da Züge die es ermöglichen drei Steine neben einander zu platzieren natürlich auch zwei Steine neben einander zur Folge haben, werden sie doppelt in das Array aufgenommen. Anschießend werden alle Züge aus *possibleSolutions* entfernt, die es dem Gegner in seinem nächsten Zug ermöglichen (über eine Diagonale oder Horizontale) zu gewinnen. Im nächsten Schritt werden alle verbleibenden Spalten, die nicht am Rand liegen ein weiteres Mal in das Array kopiert, um dann einen zufälligen Zug auszuwählen. Falls *possibleSolutions* leer ist wird nach einer Lösung gesucht, die in erster Linie erlaubt ist und dem Gegner nach Möglichkeit nicht ermöglicht, im nächsten Zug zu gewinnen.

Die Klasse *NoramlKI2* enthält außerdem eine Variable *epsilon*, die festlegt wie viel Prozent der Spielzüge rein zufällig ausgewählt werden sollen. Diese Strategie ist bekannt als epsilon-greedy-Strategie und soll eine größere Varianz an Spielzügen ermöglichen.