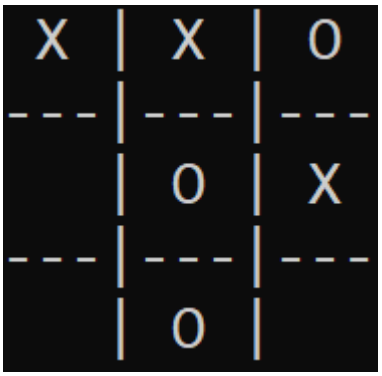
Aufgabe 4: Tic Tac Toe

In diesem Praktikum wollen wir Tic-Tac-Toe spielen[1].



Ihre Aufgabe ist es ein Programm zu schreiben, bei dem ein(e) Benutzer*in gegen den Computer spielt.

Sie müssen sich eigenständig überlegen, wie Sie ihr Programm aufbauen und welche Funktionen Sie benötigen. Vermeiden Sie die Dopplung von Programmcode, indem Sie Code in Funktionen kapseln.

Folgende Anforderungen soll ihr Programm erfüllen:

Anf-01	Der/die Benutzer*in hat den ersten Zug.
Anf-02	Für das Tic-Tac-Toe-Spielfeld soll ein zweidimensionaler Vektor (Array) verwendet werden. Vermeiden Sie eine globale Variable.
Anf-03	Die Benutzerführung ist ohne Erklärung verständlich und der/die Benutzer*in weiß bei jedem Schritt, welche Eingabe gefordert ist.
Anf-04	Die Benutzereingabe wird auf Gültigkeit überprüft.
Anf-05	Bei falscher Eingabe muss die Eingabe wiederholt werden.
Anf-06	Die gewünschte Position im Spielfeld kann in zwei Schritten vom Benutzer*in abgefragt werden (Abfrage der Zeile, dann Abfrage der Spalte).
Anf-07	Das leere Spielfeld wird am Anfang eines Spiels ausgegeben.
Anf-08	Das Spielfeld wird nach einem Zug aktualisiert.
Anf-09	Folgende Symbole werden verwendet: Benutzer*in: X Computer: O
Anf-10	Nach jedem Zug wird überprüft, ob es eine(n) Gewinner*in gibt. Falls ja, wird eine entsprechende Ausgabe ausgegeben und das Spiel beendet.
Anf-11	Die Spielstrategie des Computers ist zufällig. D.h. als nächster Zug soll das Feld zufällig aus den noch verfügbaren Feldern ausgewählt werden.
Anf-12	Falls keine Züge mehr möglich sind und keiner gewonnen hat, geht das Spiel unentschieden aus, und eine entsprechende Ausgabe erscheint.
Anf-13	Zum Testen bereiten Sie bitte vordefinierte Spielabläufe vor und notieren Sie sich hierfür die einzelnen Züge. Min. ein Spielablauf für 1. Computer gewinnt 2. Benutzer*in gewinnt

[1] Falls Sie das Spiel und seine Regeln nicht kennen, können Sie z.B. bei Wikipedia dies nachlesen: https://de.wikipedia.org/wiki/Tic-Tac-Toe

	3. Benutzer*in gewinnt im letzten Zug 4. unentschieden Setzen Sie hierfür den Seed des Zufallsgenerators mit *srand()* so, dass Sie die Spielabläufe reproduzieren können.

Solche Anforderungslisten sind sehr typisch für Software-Projekte. Allerdings ist es schwierig und aufwendig, Anforderungslisten vollständig und konsistent („ohne Widersprüche“) zu dokumentieren. Außerdem können Anforderungen zu unkonkret formuliert oder nicht überprüfbar sein.

Bitte treffen Sie Annahmen, falls nicht alle für Sie notwendigen Informationen für eine Programmierung aus der gegebenen Anforderungsliste hervorgehen. Dokumentieren Sie ihre Annahmen in einer weiteren Tabelle und erläutern Sie ihre Annahmen bei der Abnahme.

Folgende Anforderungen sind **optional** umzusetzen:

Opt-01	Es soll eine verbesserte Spielstrategie für den Computer programmiert werden. Die verbesserte Strategie ist selbst zu entwickeln.
Opt-02	Am Ende eines Spiels wird gefragt, ob man nochmal spielen möchte. Falls ja, startet das Spiel erneut.
Opt-02b	Es wird der Spielstand über mehrere Runden ausgegeben. Die Ausgabe umfasst die Anzahl für gewonnene/unentschiedene/verlorene Spiele.
Opt-03	Am Anfang wird abgefragt, mit welcher Spielstrategie der Computer spielen soll („einfach“ oder „schwer“).
Opt-04	Am Anfang wird abgefragt, wer anfängt (Benutzer*in oder Computer).