Spiel für 2–4 Spieler*innen. Jede*r Spieler*in hat einen Reinigungsroboter, den es zu programmieren gilt. Es gewinnt der Roboter, der als erstes eine Reihe zufällig ausgewählter farbiger Lurch-Steine eingesammelt hat und dann zur Ladestation zurückkehrt.

Szenario

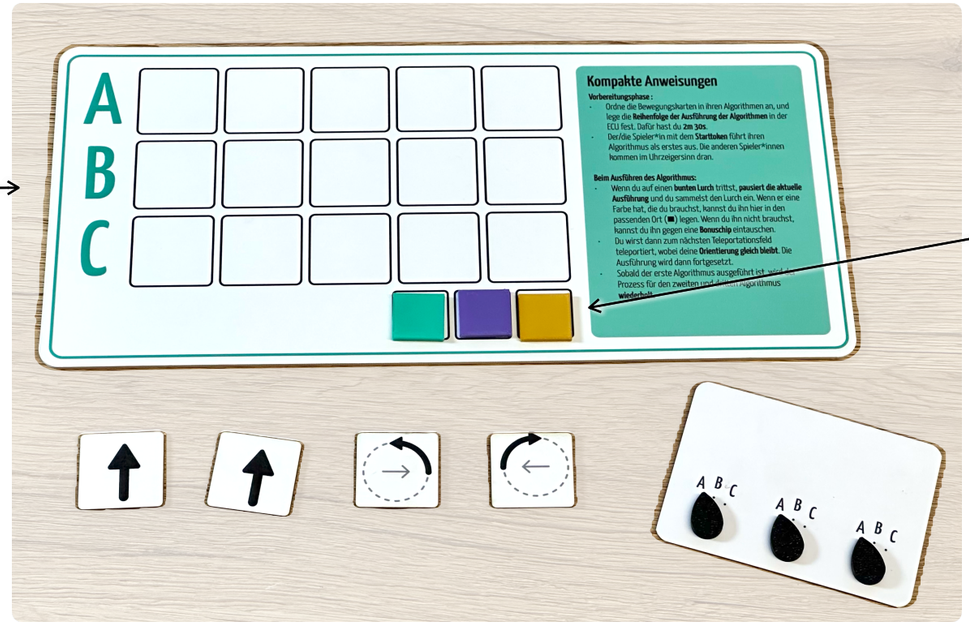
Im Jahr 2058 ist der Markt für autonome Reinigungsroboter stark umkämpft und hat sich zu regelrechten Corporate Wars entwickelt. Reinigungsroboter sind mit Hightech-Komponenten, ausgefeilter Programmierung und sogar Waffen ausgestattet, um ihre Vorherrschaft in jedem Haushalt zu sichern. Um jedes Staubkorn wird verbissen und gnadenlos gekämpft. Werden Sie Eliteprogrammierer*in und helfen Sie Ihrem Unternehmen, die Corporate Wars zu gewinnen.

Inhalt

4 Staubsaugerroboter
4 AlgoBoards
4 ECUs (Execution Control Unit)
4 Kurzanleitungen
16 ● Punkte-Steine
12 ■ Lurch-Steine
20 ?-Karten
60 Bewegungschips
54 Bonuschips
1 Starttoken

0 Vorbereitung

1. Jede*r Spieler*in bekommt einen Roboter, eine ECU (Execution Control Unit), ein AlgoBoard, und vier Bewegungschips: zwei Vorwärts-Chips und je einen Dreh-Chip nach links und rechts.



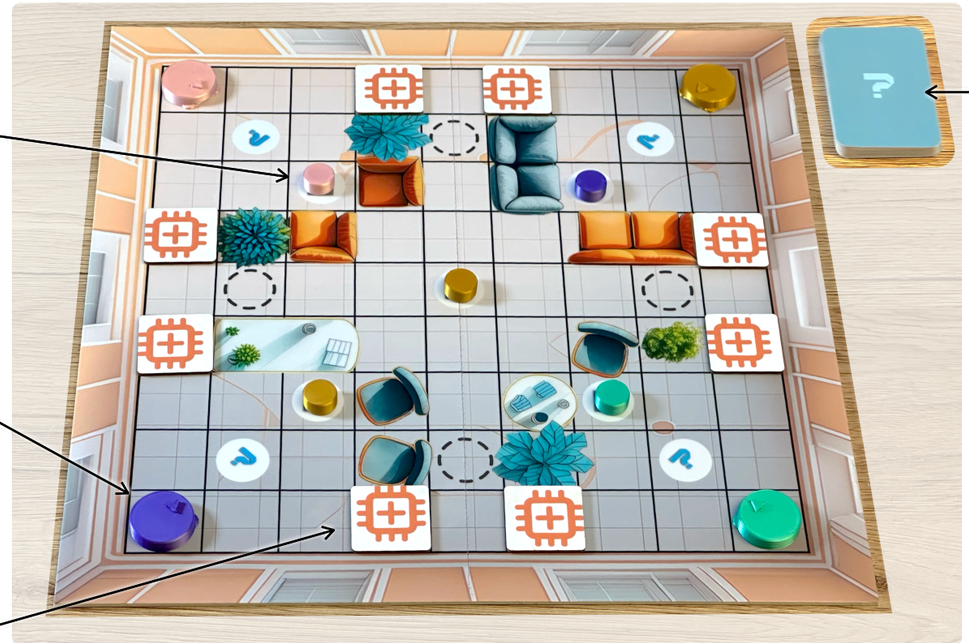
5. Jede*r Spieler*in zieht drei bunte ■ Lurch-Quadrate und legt sie auf die entsprechenden Felder auf dem eigenen AlgoBoard.

Ziel des Spiels ist es, als erstes die Punkte-Steine in diesen Farben aufzusaugen.

2. Auf die fünf ■ Lurch-Felder werden fünf zufällige ● Punkte-Steine so gelegt, dass an den Ecken jeweils eine andere Farbe liegt.

3. Die Roboter werden in den Ecken des Spielbretts auf ihre Ladestationen gestellt.

4. Es werden acht Bonuschips gezogen und verdeckt auf die + Bonusfelder gelegt.



6. Die ?-Karten werden gemischt und als Stapel neben das Spielfeld gelegt.

7. Das Starttoken geht an den*die jüngste*n Spieler*in und wandert nach jeder Runde um eine*n Spieler*in im Uhrzeigersinn weiter.

Anmerkung: Diese Spielanleitung ist unter Verwendung des * genderneutral verfasst. Sie können dieses Konstrukt gegebenenfalls beim Lesen oder Vorlesen der Anleitung gegen Formulierungen Ihrer Wahl ersetzen.

Das Spiel läuft in drei Phasen: **Programmierung, Ausführung** und **Update.**

1 Programmierung

Jede Spieler*in ordnet die Bewegungschips in den Algorithmen A, B und C neu an, und stellt die Reihenfolge der Algorithmen in der ECU ein. Mögliche Abfolgen sind A B C, A A A, C B C, etc. Dafür stehen 2m30s zur Verfügung.

Nach Ablauf der Zeit werden die Einstellungen der ECUs öffentlich gemacht.

Bewegungschips

- ein Feld in Fahrtrichtung ziehen
- 90° in die angegebene Richtung drehen
- 180° drehen
- Diagonal ziehen und gleichzeitig 90° drehen

Bonuschips

- Der Roboter platziert eine Bombe auf dem Feld, auf dem er steht. Wenn ein anderer Roboter dieses Feld betritt, explodiert die Bombe und der*die Spieler*in muss einen Punkt abgeben.
- Der Roboter schießt in Fahrtrichtung. Wenn der Schuss einen Roboter trifft, muss dieser einen Punkt abgeben.
- Der*die Spieler*in kann einen Bug auf einem beliebigen AlgoBoard platzieren. Der verdeckte Chip wird bei der nächsten Ausführung übersprungen, dann wird der Bug entdeckt und entfernt.
- x2 Der Algorithmus springt zurück an den Anfang, der Algorithmus wird ein zweites Mal ausgeführt.
- Schützt den Roboter vor Schaden durch Bomben und Schüsse.
- A Der angegebenen Algorithmus wird als Unterprogramm ausgeführt

Alle Bonuschips werden zurückgelegt, nachdem sie einmal zum Einsatz gekommen sind.

2 Ausführung

Die Chips in einer Reihe auf dem AlgoBoard definieren von links nach rechts einen Algorithmus. In dieser Phase lassen alle Roboter der Reihenfolge nach drei Algorithmen in der Reihenfolge ablaufen, die in der ECU festgelegt ist.

1. Alle Spieler*innen führen der Reihe nach den ersten auf der ECU gewählten Algorithmus aus. Dabei gilt:
   - › Wände, Möbel und Pflanzen sowie die Ladestationen anderer Spieler*innen können nicht überfahren werden.
   - › Trifft ein Roboter auf einen anderen Roboter, so schiebt er diesen vor sich her. Wenn der Roboter nicht weitergeschoben werden kann, ist er ein unüberwindbares Hindernis.
   - › Betritt ein Roboter ein Feld mit einem verdeckten Bonuschip, so wird dieser aufgesaugt und steht dem*der Spieler*in zur Verfügung.
   - › Betritt ein Roboter ein ?-Feld, kann der*die Spieler*in eine ?-Karte vom ?-Stapel abheben. Die Karte wird am Ende des Algorithmus ausgeführt.
   - › Trifft ein Roboter auf einen offenen Bonuschip, so wird dieser sofort ausgeführt und zurückgelegt.
   - › ● Punkte-Steine werden aufgesaugt. Braucht diese*r Spieler*in die Farbe, wird sie auf dem entsprechenden Lurch-Stein auf dem AlgoBoard abgelegt; braucht er*sie die Farbe nicht, kann der*die Spieler*in einen Bonus-Chip ziehen. Danach wird der Roboter zu einem der nächsten Teleport-Felder gebeamt, wo die Ausführung des Algorithmus fortgesetzt wird.
   - › Tritt ein Roboter auf eine Bombe oder wird er von einem Schuss getroffen, verliert der*die Spieler*in einen Punkt-Stein.
2. Alle Spieler*innen führen der Reihe nach den zweiten auf der ECU gewählten Algorithmus aus. Es gelten dieselben Regeln wie bei Schritt 1.
3. Alle Spieler*innen führen der Reihe nach den dritten auf der ECU gewählten Algorithmus aus. Es gelten dieselben Regeln wie bei Schritt 1

3 Upgrade

1. Jede*r Spieler*in zieht einen Bewegungschip.
2. Alle leeren ■ Lurch-Felder werden mit zufällig gewählten ● Punkt-Steinen aufgefüllt.
3. Alle leeren + Bonusfelder werden mit zufällig gewählten Bonus-Chips aufgefüllt, auch wenn diese dann unter einem Roboter liegen.
4. Das Starttoken wird an den*die nächste Spieler*in weitergegeben.

Spielende

Das Spiel endet, wenn der erste Roboter die drei Lurchsteine auf dem AlgoBoard mit Punktesteinen in derselben Farbe ergänzt hat.

Robo Rumble wurde von Jörg Artaker, Pau Ros Gimeno und Max Irendorfer 2024 gestaltet und mit Mitteln des Vizerektorats für Lehre produziert. Teile der Spielmaterialien wurden mittels genAI produziert.

Übersetzung, Redesign, Layout und Produktion Tala ElMouassarani, Naemi Luckner, Michael Pollak und Peter Purgathofer

Sie können die Spielmaterialien von Robo Rumble auch kaufen: https://piglab.org/brettspiele