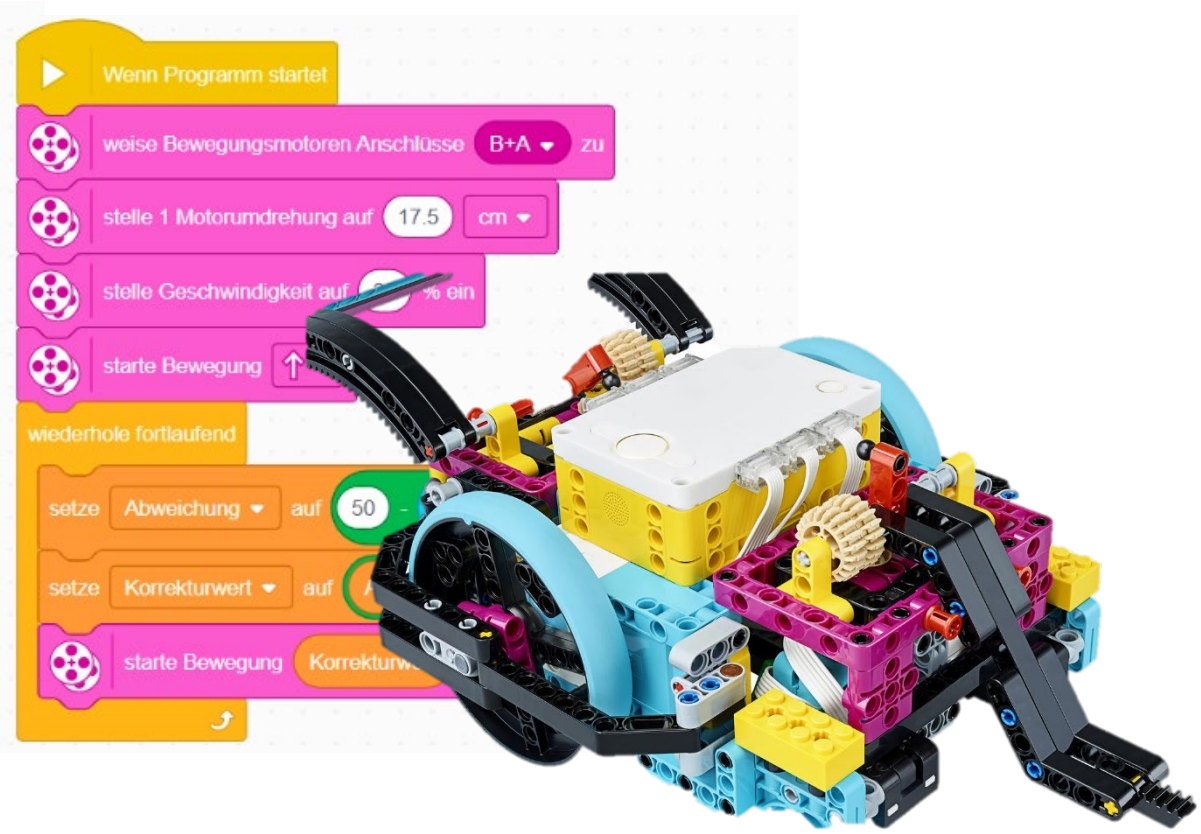
SWITZERLAND

Robotik für Anfängerinnen und Anfänger

Lerndossier

Mit didaktischem Kommentar für Lehrpersonen



von Vera Hausherr
Version 2.0 /11.09.2023

Copyright

Markenzeichen

Technischer Hinweis

Dieses Lerndossier wurde auf der Basis der Version 3 der SPIKE-App verfasst. Es kann kleine Abweichungen zwischen den Abbildungen in diesem Dossier und früheren (oder noch unveröffentlichten späteren) Versionen der SPIKE-App geben.

Lust auf mehr Robotik?

Dann mach mit am Wettbewerb der World Robot Olympiad! Alle Informationen findest du auf https://wro.swiss

Teil 1: Was ist Robotik?

1: Maschinen, Computer und Roboter

Lernziel:

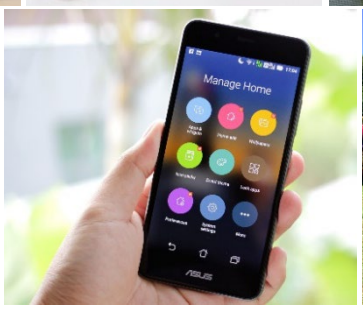
- ☐ Ich kenne den Unterschied zwischen Maschinen, Computern und Robotern.

Trage diese Gegenstände richtig in die Tabelle ein. Fallen dir auch noch andere Beispiele ein?







Maschine	Computer	Roboter
Bagger	*Laptop*	*Staubsaugerroboter*
Küchenmaschine	*Handy*	*Serviceroboter*
Traktor	*z.B. Geldautomat*	*Industrieroboter*
z.B. Nähmaschine	*z.B. Game-Konsole*	*Spike Prime Legoroboter*
z.B. Achterbahnr	*z.B. SmartHome-Gerät*	*z.B. Fussballroboter*

Merke:
Eine **Maschine** kann arbeiten, aber nicht denken.
Ein **Computer** kann denken, aber nicht arbeiten.
Ein **Roboter** kann arbeiten und denken.

Didaktischer Kommentar

Bei dieser einfachen Aufgabe geht es darum, dass die Schülerinnen und Schüler den Unterschied zwischen Maschinen, Computern und Robotern erkennen. Das wichtigste Merkmal eines Roboters ist, dass er mittels Sensoren seine Umwelt wahrnehmen und entsprechend darauf reagieren kann. Da gibt es einfache Roboter wie den Rasenmähroboter oder einen Industrieroboter, aber es gibt auch komplexere Roboter wie etwa solche, die in der Altenpflege oder der Katastrophenrettung eingesetzt werden. Diese müssen sehr viele unterschiedliche Reize wahrnehmen und analysieren können und dann richtig entscheiden. Es gibt auch Roboter, die Fussball spielen können (dafür gibt es sogar eine eigene WM!), und dann natürlich die selbstfahrenden Autos.

Weiterführende Ideen

- *Recherchearbeit zu unterschiedlichen Robotern: Es gibt sehr interessante Videos zu Robotern, die schier unglaubliches leisten.*
- *Im Fächerverbindenden Unterricht: Kreatives Schreiben zum Thema: Eine Welt voller Roboter.*
- *Eine Suche nach «Boston Dynamics» auf YouTube liefert viele Treffer. Boston Dynamics ist eine Firma, die in diesem Bereich zu den besten der Welt gehört – allerdings ist sie ursprünglich im Rüstungssektor angesiedelt, und viele Roboter, die einen zivilen Nutzen haben können, wurden ursprünglich für militärische Zwecke angesiedelt. Insbesondere bei grösseren Kindern könnte das auch Anlass für eine ethische Diskussion sein.*
- *Im Werken-Unterricht: einen Prototyp für einen Roboter bauen, der eine bestimmte Aufgabe erledigen kann. Je nach Alter der Schülerinnen und Schüler kann dies eine einfache Bastelarbeit sein, bei der das Produkt einfach aussieht wie ein Roboter, oder eine (mechanische) Maschine, die sich tatsächlich bewegen und etwas tun kann.*

2: Was für Teile braucht ein Roboter?

Lernziele:

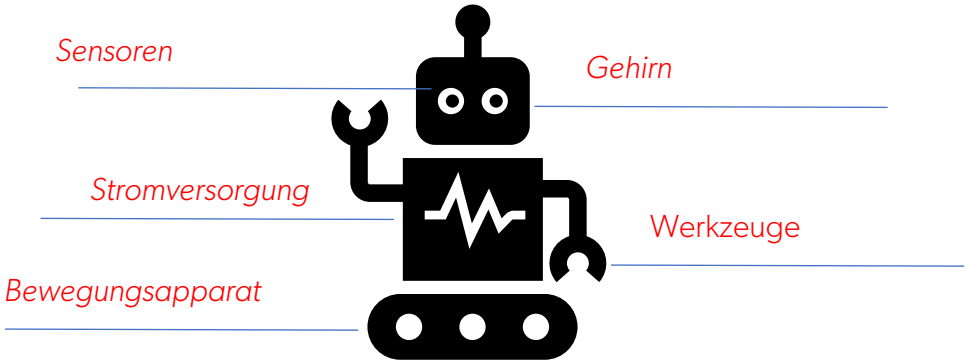
- ☐ Ich weiss, welche Teile zu einem Roboter dazu gehören.
- ☐ Ich weiss, warum Sensoren wichtig sind.
- ☐ Ich kenne die Bestandteile des SPIKE Prime.

In der vorherigen Aufgabe hast du gesehen, dass es ganz unterschiedliche Roboter gibt. Aber all diese Roboter haben bestimmte Teile gemeinsam.

2.1 Übersicht

Aufgabe:

Ordne diese Begriffe dem Roboter zu: *Bewegungsapparat*, *«Gehirn»*, *Sensoren*, *Stromversorgung*, *Werkzeuge*.



2.2 Wofür braucht man diese Bestandteile?

Aufgabe:
Trage die Begriffe von der Aufgabe 2.1 in die Tabelle ein.

Wahrnehmung der Umwelt (z.B. eine Farbe erkennen oder einen Abstand messen)	*Sensoren*
Fortbewegung (z.B. Laufen oder Fahren)	*Bewegungsapparat*
Steuerung des Roboters (Entscheiden, was er als nächstes tun muss)	*Gehirn*
Arbeiten (z.B. einen Gegenstand einsammeln)	*Werkzeuge*
Energie zum Fahren und Arbeiten haben	*Stromversorgung*

2.3 Welche Teile gibt es beim SPIKE Prime?

Aufgabe:
Ergänze die Tabelle.
Benötigte Wörter: *Bewegungen, dunkel, entfernt, Fahren, Farben, «Gehirn», hell, hochzuheben, Knöpfe, Lichtmatrix, Schalter, weit.*

	**Hub mit Akku und Gyrosensor**	Der Hub enthält das *Gehirn* des Roboters. Auf der weissen Oberfläche sind die *Lichtmatrix* und *Knöpfe*. Mit dem Gyrosensor kann man erkennen, ob der Roboter gedreht oder gekippt wurde.
	**Motoren**	Motoren braucht man für alle *Bewegungen*. Die kleinen Motoren sind gut zum *Fahren*, die grossen sind gut, um etwas *hochzuheben*.
	**Farbsensoren**	Mit Farbsensoren kann man *Farben* erkennen und messen, wie *hell* oder *dunkel* etwas ist.
	**Abstandssensor**	Mit dem Abstandssensor kann man messen, wie *weit* ein Gegenstand, ein Hindernis oder eine Wand _____ ist.
	**Kraftsensor**	Mit dem Kraftsensor kann man erkennen, wie fest er gedrückt wird. Man kann ihn benutzen wie einen *Schalter*.

Didaktischer Kommentar

Die Bestandteile des Roboters kann man entweder an dieser Stelle einführen oder später bei den «Ersten Schritten» Der SPIKE App. Es ist sicher sinnvoll, zu thematisieren, warum Sensoren so wichtig sind. Eine gute Kenntnis der Sensoren und ihrer Möglichkeit ist unumgänglich, um eine anspruchsvolle Aufgabe, wie etwa eine WRO Wettbewerbsaufgabe, zu lösen.

Weiterführende Ideen:

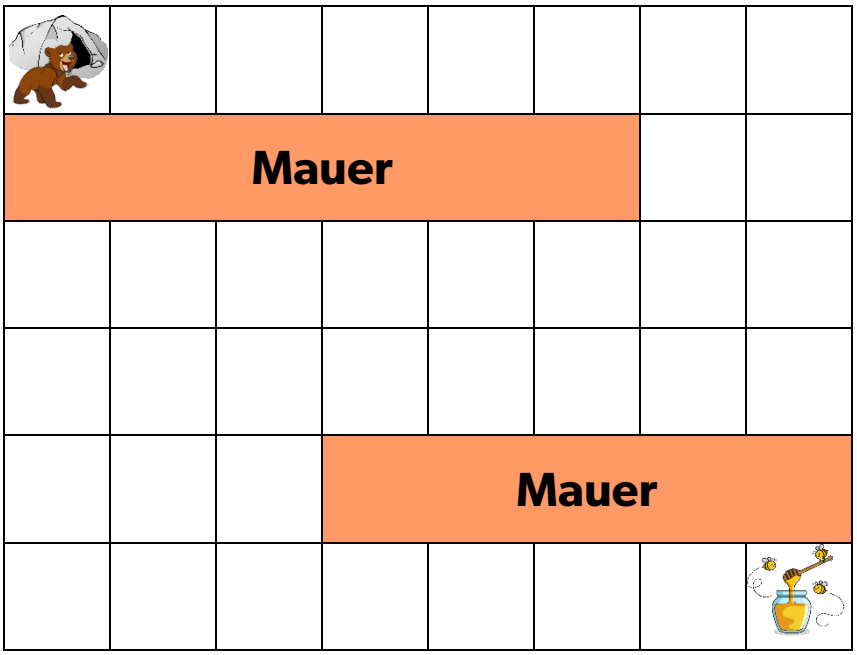
- *Fächerverbindender Unterricht mit Biologie: Vergleich Sensoren / Sinnesorgane der Menschen.*

3: Woher weiss der Roboter, was er arbeiten muss?

Lernziele:

- ☐ Ich kenne die Begriffe «Programmieren» und «Code».
- ☐ Ich kann einfache Anweisungen für einen Roboter in normaler Sprache geben.

Der «Bären-Roboter» soll von seiner Höhle zum Honigtopf kommen. Er kann nicht über die Mauern klettern. Und er kann sich über das Feld nur von links nach rechts, von rechts nach links, von oben nach unten und von unten nach oben bewegen (nicht schräg.)



Aufgabe:

1. Zeichne den besten Weg in den Plan ein.
2. Erkläre deinem Partner/deiner Partnerin, wie der Bär am besten zum Ziel kommt.

Merke:
Damit dein Bären-Roboter den Weg findet, musst du ihm klare Anweisungen geben.

Das nennt man *programmieren*.

Solche Anweisungen nennt man *Code oder Programm*.

Aufgabe: Ergänze diesen Code, damit der Bären-Roboter den Weg von der Höhle zum Honigtopf findet.
(Mögliche Wörter: «rechts», «links», «oben», «unten» und die Zahlen von 0 bis 12.)

- Verlasse die Höhle nach *rechts*
- Gehe *6* Felder geradeaus.
- Drehe dich nach *unten*.
- Gehe *2* Felder geradeaus.
- Drehe dich nach *links*.
- Gehe *4* Felder geradeaus.
- Drehe dich nach *unten*.
- Gehe *3* Felder geradeaus.
- Drehe dich nach *rechts*.
- Gehe *5* Felder geradeaus.

Didaktischer Kommentar

Ein Roboter ist schlau und kann denken, aber er kann nur das denken, was ihm vorher von einem Programmierer oder einer Programmiererin beigebracht wurde. Wenn man sich die Einzelanweisungen im Code anschaut, dann kommt man zum Schluss, dass der Roboter eigentlich sogar ziemlich dumm ist. Die Übung mit dem Bärenroboter und dem Honigtopf kann selbst ein kleines Kind leicht erledigen, aber ein Roboter muss genau gesagt bekommen, was er tun soll.

Weiterführende Ideen

- *Die Einschränkung, dass der Roboter sich nur horizontal und vertikal über das Spielfeld bewegen kann, ist natürlich nicht in Stein gemeisselt, sondern eine willkürliche Vorgabe für diese Übung. Man könnte andere Einschränkungen oder Möglichkeiten dazu nehmen, z.B. dass der Roboter auch diagonal fahren darf, oder dass eine Mauer hoch ist und umfahren werden muss, und eine Mauer niedrig ist und überklettert werden kann. Wie sieht dann der Code aus?*

Es gäbe für die Lösung der Aufgabe auch noch andere Möglichkeiten, die allerdings fortgeschrittenere Programmierfähigkeiten voraussetzen. Z.B. könnte man dem Roboter einen Abstandssensor an der Seite anbauen, und er soll immer in einem bestimmten Abstand zur Wand fahren, und wenn die Wand einen Knick macht, diesem Knick folgen.

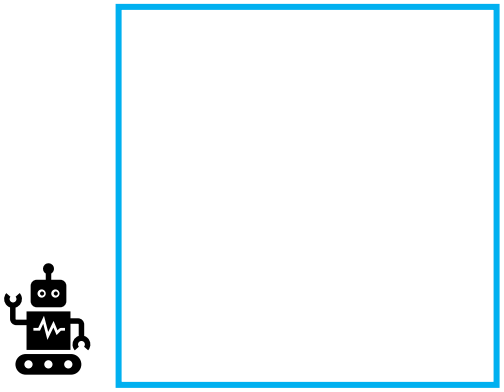
Teil 2: Einführung ins Programmieren

4: Den Code etwas vereinfachen: Schleifen

Lernziel

- ☐ Ich weiss, was eine Schleife ist und wofür man sie braucht.

Dein Roboter soll einen Weg laufen, der genau ein Quadrat ergibt.



Aufgabe:

1. Dein Partner/deine Partnerin ist der Roboter. Gib ihm/ihr Anweisungen, wie er/sie das Quadrat laufen soll. Eine Seite des Quadrats soll dabei genau 5 Schuhe lang sein.
2. Versuche, eine sehr kurze Anleitung zu schreiben. Sie soll genau aus 3 Zeilen Code bestehen.

Gehe 5 Schuhlängen geradeaus

Drehe dich nach rechts

Wiederhole das ganze 4 Mal

Gehe 5 Schuhlängen geradeaus

3. Jetzt schreiben wir den Code so ähnlich, wie er dann bei unserem Lego-Roboter aussehen wird: (Dies ist noch kein echter Code, er sieht nur so aus!)

Überlege: Warum ist es nützlich, solche Wiederholungen zu verwenden?

Merke:

Einen Wiederholungs-Block nennt man «**Schleife**».

Und hier ist noch eine goldene Programmier-Regel für dich: **Schreibe nie etwas zweimal, wenn du es auch einmal schreiben und dann eine Schleife verwenden kannst!**

Didaktischer Kommentar

Beim Programmieren lautet eine goldene Regel:

Schreibe nicht etwas zweimal, wenn einmal auch reicht.

Dies hat mehrere Gründe:

Historisch gesehen, war Speicherplatz früher knapp, weshalb man keinen Platz verschwenden wollte. Jede Zeile, die man einsparen konnte, war kostbar. Viel wichtiger ist jedoch die Fehlervermeidung: Wenn man z.B. den Code für dieses Quadrat in der Langform schreiben würde und jede Seitenlänge, jede Ecke als eigene Codezeile, dann würden sich leicht Fehler einschleichen. Es wäre etwa möglich, dass man bei einer Ecke rechts und links verwechselt. Dann kommt natürlich kein Quadrat mehr heraus, sondern eine komische Form. Wenn man mit Schleifen arbeitet, dann programmiert man einmal den Inhalt der Schleife, testet, und wiederholt dann das, was funktioniert.

Es gehört dazu zu den absoluten Grundfähigkeiten eines Programmierers oder einer Programmiererin zu erkennen, welche Aufgabenteile sich wiederholen.

Weiterführende Ideen:

- *Wenn die Schülerinnen und Schüler die «ersten Schritte» mit der SPIKE App (Kapitel 5) gemacht haben, können sie selbst Aktionen erfinden, die sie wiederholen. Z.B. könnten sie mit den Klang-Blöcken eine einfache Melodie schreiben und diese zweimal abspielen, oder sie könnten eine Kombination aus Licht- und Klangeffekten mehrmals abspielen. Es ist sinnvoll, immer wieder auf den Nutzen der Wiederholung hinzuweisen.*

Das Prinzip «Schreibe nicht etwas zweimal, wenn einmal auch reicht» ist bei anspruchsvolleren Aufgaben nicht nur für unmittelbare Wiederholungen nützlich, sondern auch dann, wenn die identische Aufgabe mehrfach durchgeführt werden muss. In einer WRO-Wettbewerbs-Aufgabe könnte das z.B. das Scannen und Einsammeln eines Objekts sein. Für eine solche Situation kann man den mehrfach benötigten Code in einen «Eigenen Block» (MyBlock) auslagern, der dann nur einmal programmiert und beliebig oft aufgerufen werden kann.

4: Kann der Roboter auch etwas intelligenter sein? Bedingungen

Lernziele

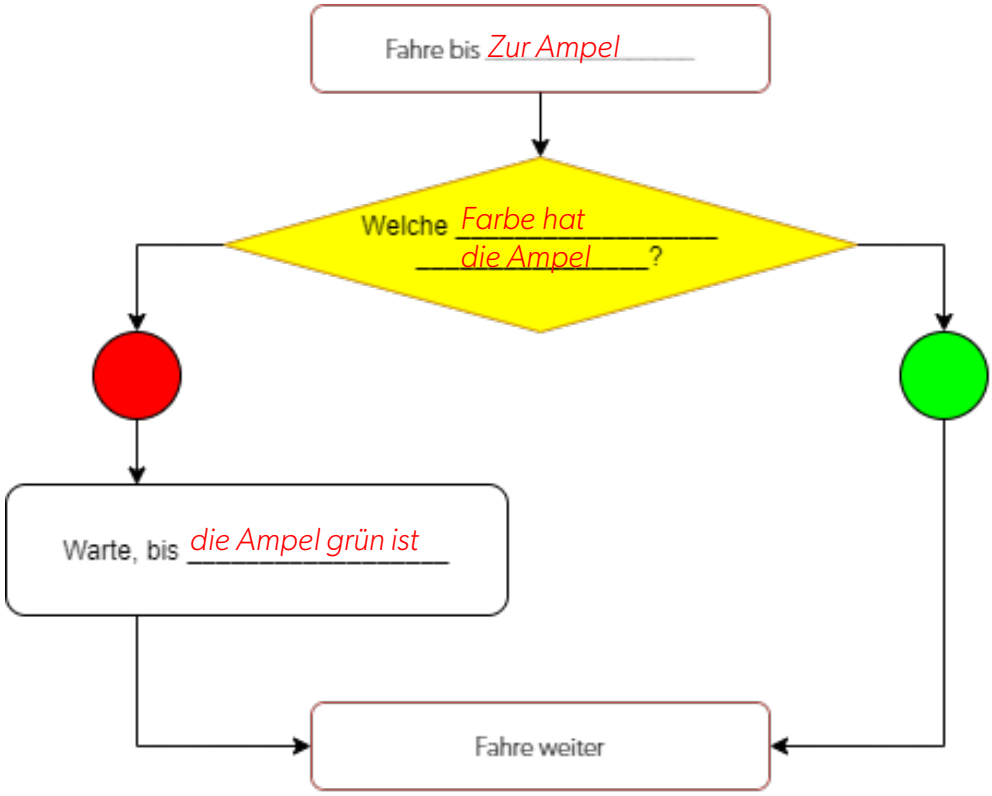
- ☐ Ich weiss, dass ein Roboter immer wieder Entscheidungen treffen muss.
- ☐ Ich kann eine einfache Bedingung erklären.

Mit einfachem Code wie beim Bären-Roboter oder beim Quadrat kann ein Roboter schon einige gute Sachen machen. Aber er muss eigentlich noch nicht selbst denken. Wir wollen aber einen wirklich schlauen Roboter, der mitdenkt und selbst entscheidet, was er tun muss.

Beispiel: Dein Autoroboter im Bild oben soll fahren. Wenn er an eine Ampel kommt, muss er an der gelben Linie überprüfen, welche Farbe die Ampel hat. Wenn sie rot ist, muss er stehen bleiben, bis die Ampel grün ist. Wenn die Ampel grün ist, muss er nicht anhalten.

Aufgabe:

Ergänze den Code, der hier als Flussdiagramm dargestellt ist:



Solche Situationen nennt man **Entscheidungen** oder **Bedingungen**. Sie sind beim Programmieren das Wichtigste!

Didaktischer Kommentar

Entscheidungen sind das, was wirklich den Unterschied zwischen einer Maschine und einem Roboter ausmacht. Der Roboter nimmt mit Sensoren seine Umwelt wahr und entscheidet dann, wie er sich verhalten muss.

Auch eine Modelleisenbahn kann fahren, aber sie kann nicht selbst entscheiden, ob sie halten oder weiterfahren soll, das muss der Modelleisenbahnbauer entweder manuell schalten, oder so bauen, dass in bestimmten Situationen der Loki einfach der Strom abgestellt wird und sie nicht weiterfahren kann. Bei grossen Anlagen kann das durchaus computergesteuert sein, aber damit wird aus der Loki noch kein Roboter.

Ein Roboter hingegen kann (im Rahmen seiner technischen Möglichkeiten) ***autonom*** *entscheiden, ob er in bestimmten Situationen Handlung 1 oder Handlung 2 durchführen muss.*

Deshalb ist es wichtig, dass die Schülerinnen und Schüler sich immer wieder überlegen: Welche Entscheidung ist hier notwendig? Auch dann, wenn die Aufgabe trivial erscheint. Die Wettbewerbsaufgaben der WRO zum Beispiel sind gerade für die Altersgruppe Elementary zumeist so konzipiert, dass die Lösung für uns Menschen sehr einfach erscheint; oft können wir auf einen Blick sehen, was zu tun ist. Der Roboter hingegen kann immer nur einen sehr begrenzten Teil der Wirklichkeit auf einmal wahrnehmen: Einen Weg, eine Querlinie zum Abbiegen, eine Linie zum Anhalten, die Farbe eines Objekts… und bei jeder Station sind neue Entscheidungen nötig.

Wenn dann später noch Algorithmen dazu kommen (vgl. Teil 4), dann werden Entscheidungen praktisch zum Dauerzustand. Beim Linienverfolger, bei dem konstant (d.h. bis zu 100 mal pro Sekunde) die Helligkeit des Untergrunds gemessen und die Fahrtrichtung entsprechend korrigiert wird, ist das exemplarisch zu erkennen: Die Bedingung ist eine bestimmte Helligkeit, die Entscheidung ist, in welche Richtung korrigiert werden muss.

Weiterführende Ideen:

- *Häufige Alltagshandlungen daraufhin untersuchen, welche Entscheidungen bei ihnen stattfinden. Beispiel: Habe ich Hunger? → essen, Bin ich müde? → schlafen gehen, kommt ein Auto? → am Strassenrand warten, ist morgen eine Prüfung? → heute noch lernen… Auf diese Art erkennen die Schülerinnen und Schüler, wie häufig Entscheidungen tatsächlich notwendig sind und werden für die Wichtigkeit von Entscheidungen fürs Programmieren sensibilisiert.*

- *Die Alltagsentscheidungen in einem Flussdiagramm darstellen. Dieses kann auch mehrere Stufen umfassen:*
  Habe ich Hunger? → Habe ich grossen Hunger? → Habe ich Lust auf etwas Süsses? → Mag ich lieber Gummibärli oder Schokolade? ….

Teil 3: Ein Spike Prime Roboter

5: Die ersten Schritte mit dem SPIKE Prime

Lernziele

- ☐ Ich kenne die Bauteile des SPIKE Prime.
- ☐ Ich kenne die SPIKE App.
- ☐ Ich kann schon ersten Code selbst schreiben.

Jetzt geht es ans Bauen und Programmieren von deinem Roboter!

Starte auf deinem Computer oder Tablet die **SPIKE App** und klicke dann auf **Prime**.

Mit dem Abschnitt **«Erste Schritte»** geht es los.

Es gibt dort 6 Aufgaben, mit denen du den Hub, die Motoren und Sensoren kennenlernen und erste Schritte beim Programmieren machen kannst.

Aufgabe:

Arbeite alle 6 Aufgaben durch. Wenn du schnell fertig bist, kannst du noch folgende Varianten versuchen:

Zusatzaufgabe 1 (Die Lichtmatrix): Kannst du auch einen Pfeil machen? Wie kannst du alle Lichter an oder ausschalten?

Zusatzaufgabe 2 (Der Motor): Kannst du den Motor auch andersrum laufen lassen?

Zusatzaufgabe 3 (Der Farbsensor): Probiere es noch mit anderen Farben aus. Nimm Legosteine zum Testen und andere Gegenstände, die ähnliche Farben haben. Probiere auch aus, wie nah der Gegenstand am Sensor sein muss, damit er wirklich gut erkannt wird.

Zusatzaufgabe 4 (Der Abstandssensor): Kannst du den Abstand auf 15 cm einstellen? Kannst du einen anderen Ton abspielen?

(Es gibt keine **Zusatzaufgabe 5**, da im Tutorial zu wenig Befehle zur Verfügung stehen!)

Zusatzaufgabe 6 (Der Gyrosensor): Kannst du auch messen, ob der Hub nach vorne gekippt ist? Und kannst du auch andere Geräusche abspielen? Probiere es aus!

Didaktischer Kommentar

In diesem Abschnitt geht es darum, dass die Schülerinnen und Schüler die Komponenten des SPIKE Prime Roboters kennenlernen. Die «Ersten Schritte» in der App sind selbsterklärend. Mit den weiterführenden Aufgaben können die Kinder noch einige Sachen ausprobieren, die später nützlich sind. Dies ist auch im Sinne der Binnendifferenzierung sinnvoll, da es ja immer einzelne Kinder gibt, die besonders schnell fertig sind, während andere noch Hilfe benötigen.

Die Aufgaben in diesem Abschnitt sind auch noch eine gute Übung zum Thema Bedingungen. Allerdings sind das hier noch die Bedingungen, die nicht in einen Gesamtablauf eingebettet sind. Bei dieser Form von Bedingungen sitzt der Roboter untätig herum und wartet, bis eine Bedingung erfüllt ist, bevor er etwas tut. Wenn man die Bedingung in einen Ablauf einbetten will, braucht man den Wenn-Dann-Block oder den Wenn-Dann-Sonst-Block.

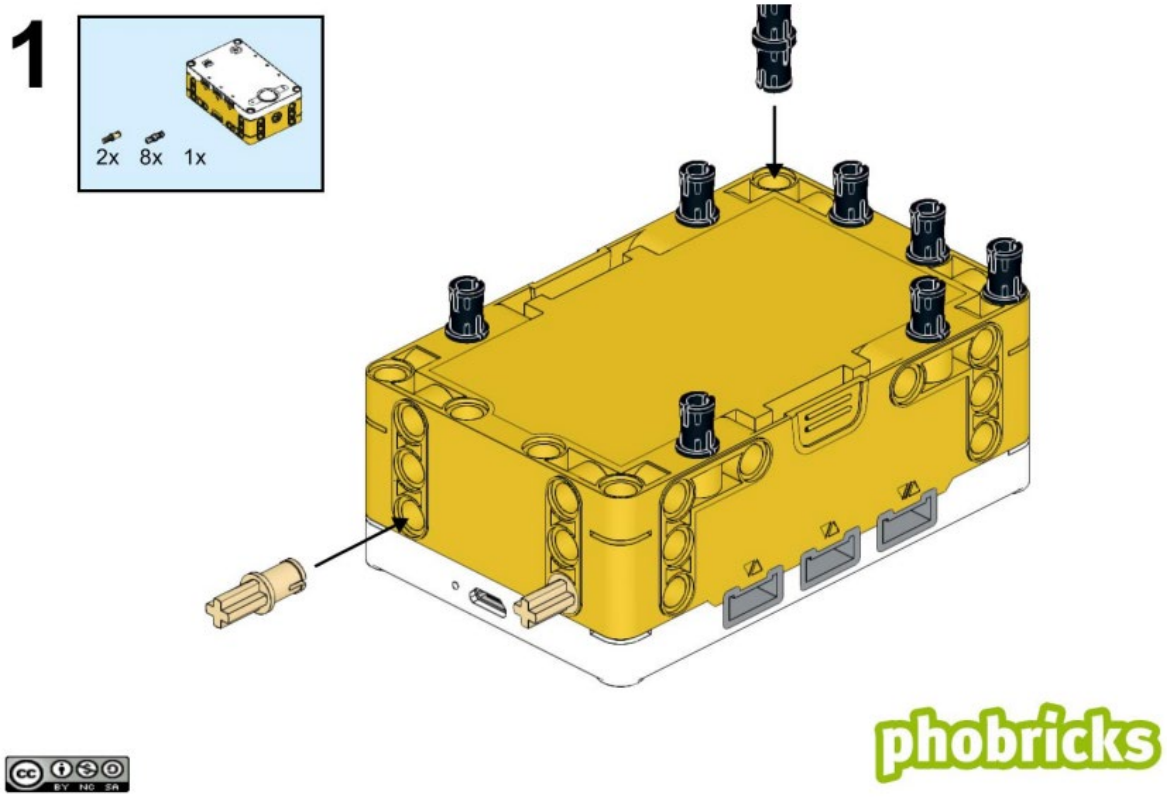
Weiterführende Ideen:

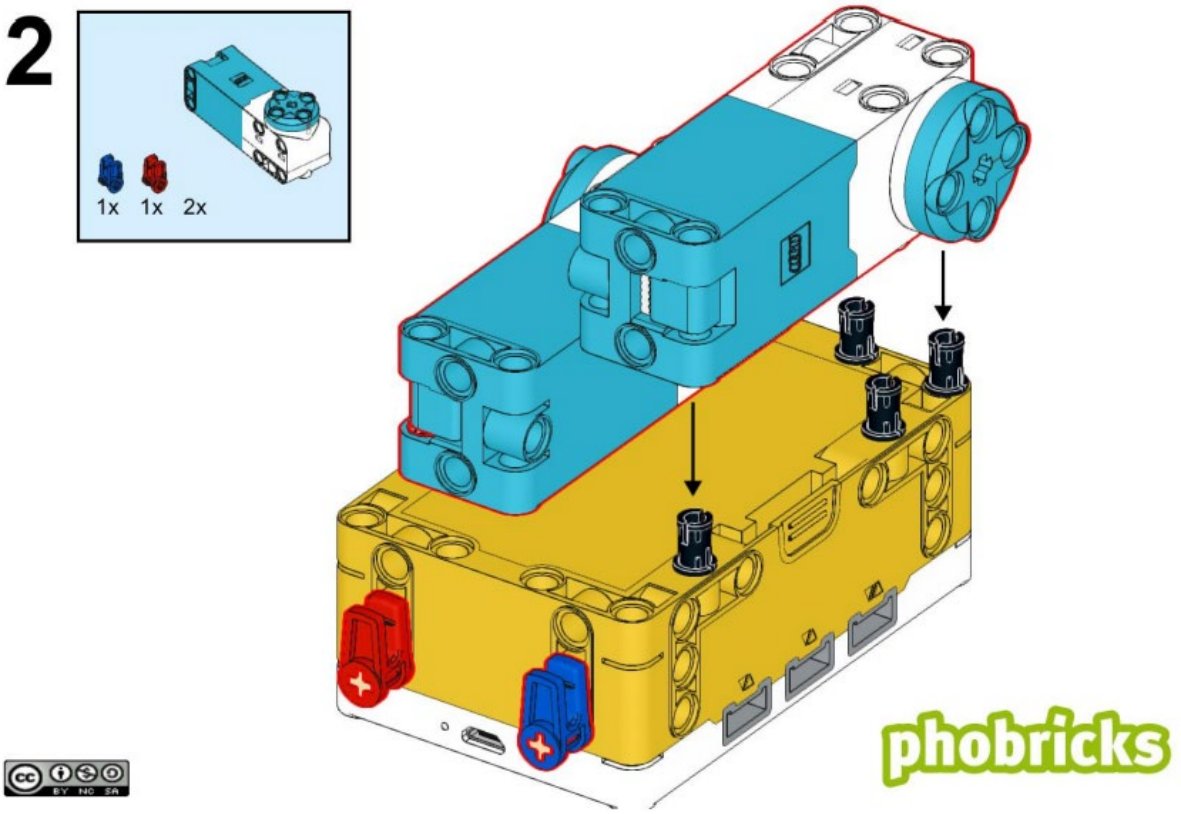
- *Die Schülerinnen und Schüler könnten sich weitere Aktionen mit den verschiedenen Komponenten ausdenken und diese auch kombinieren.*

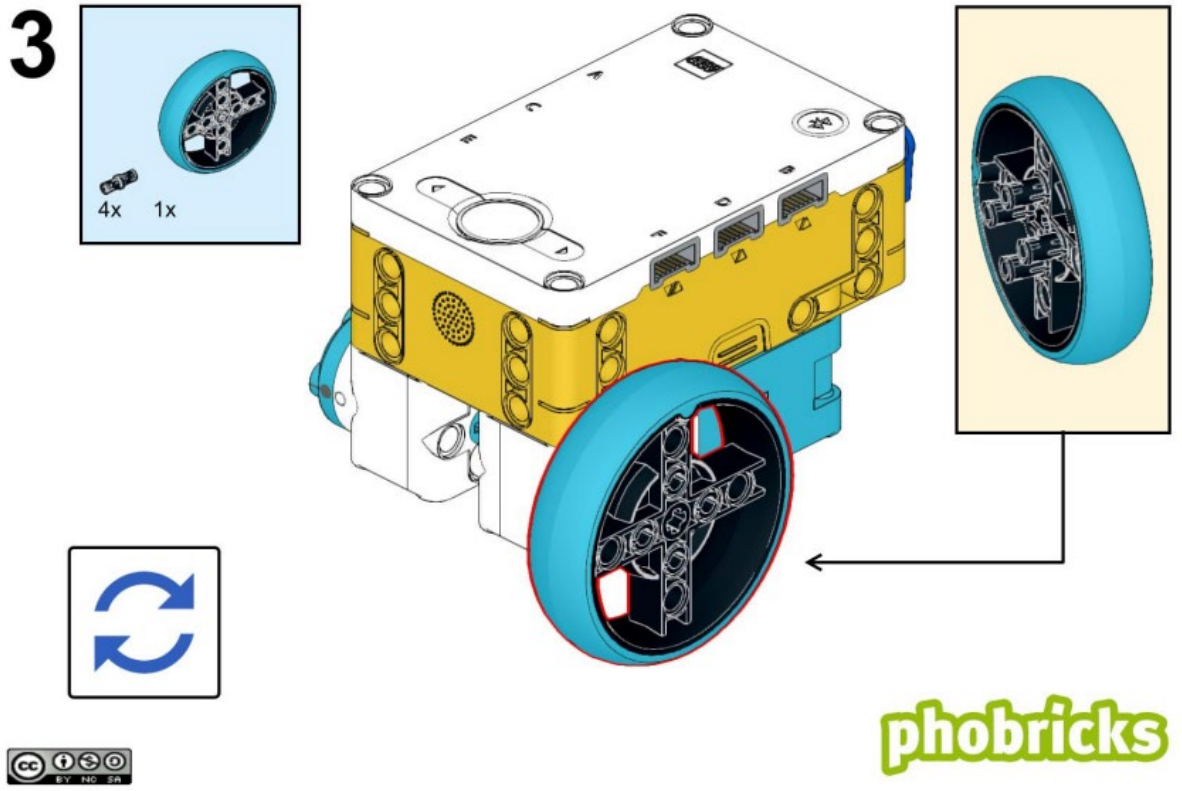
6: Baue einen kleinen Roboter.

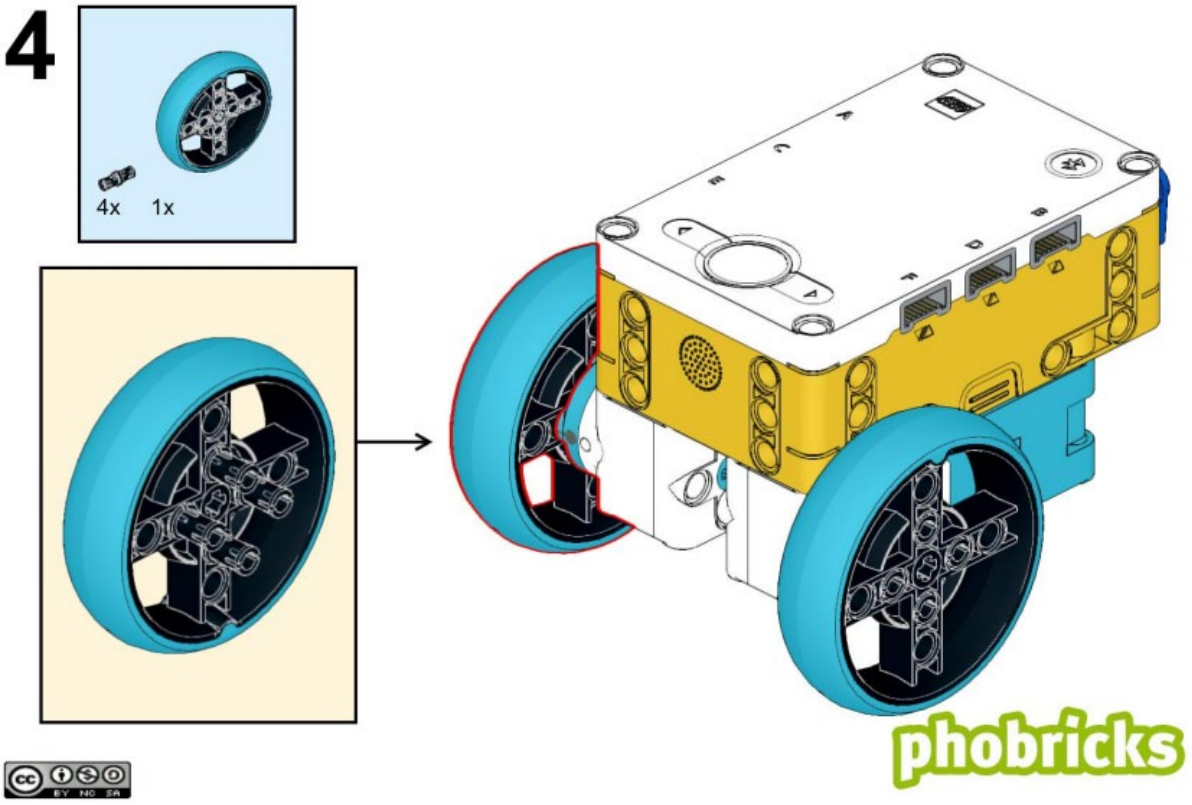
Lernziel

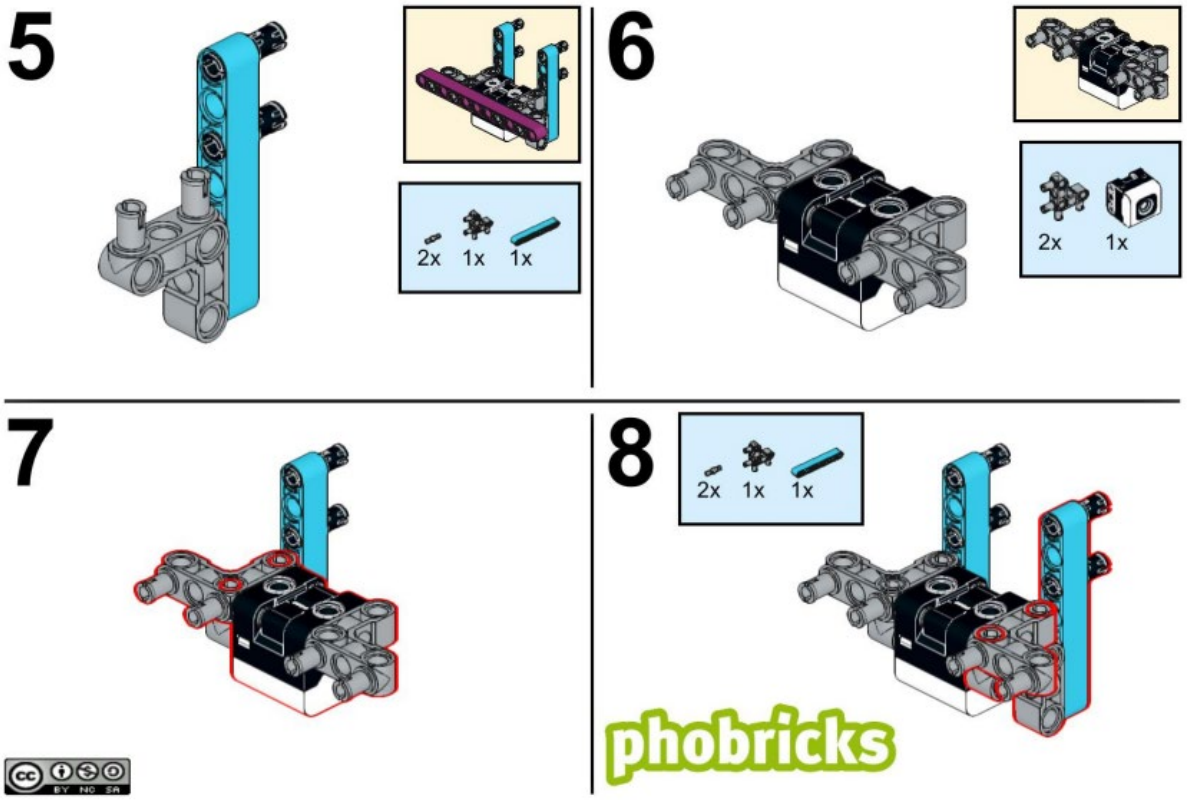
- ☐ Ich baue einen kleinen Roboter mit Farbsensor.

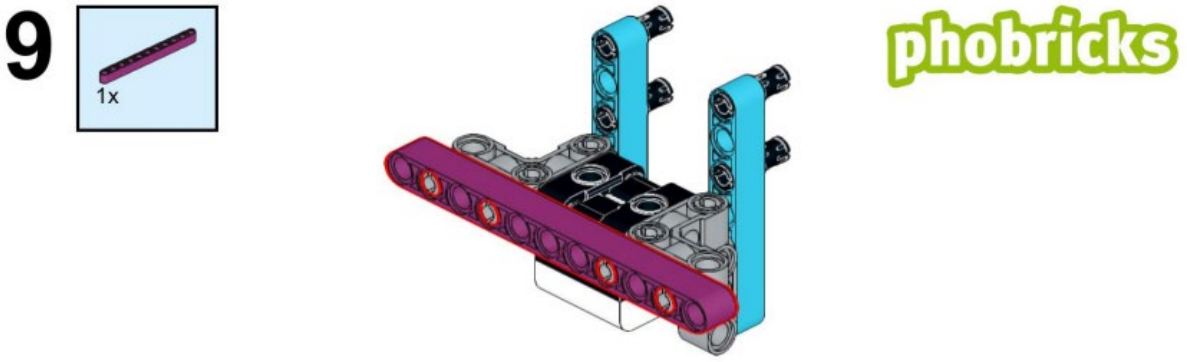


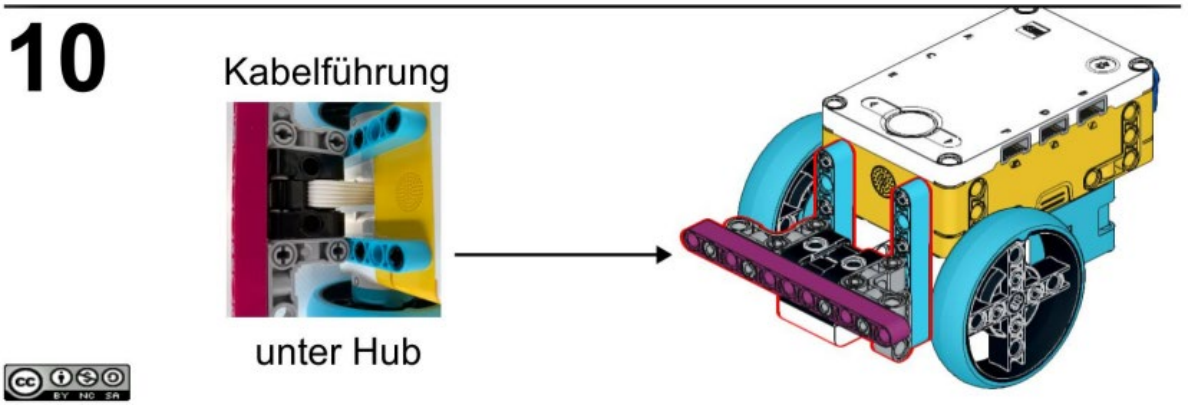


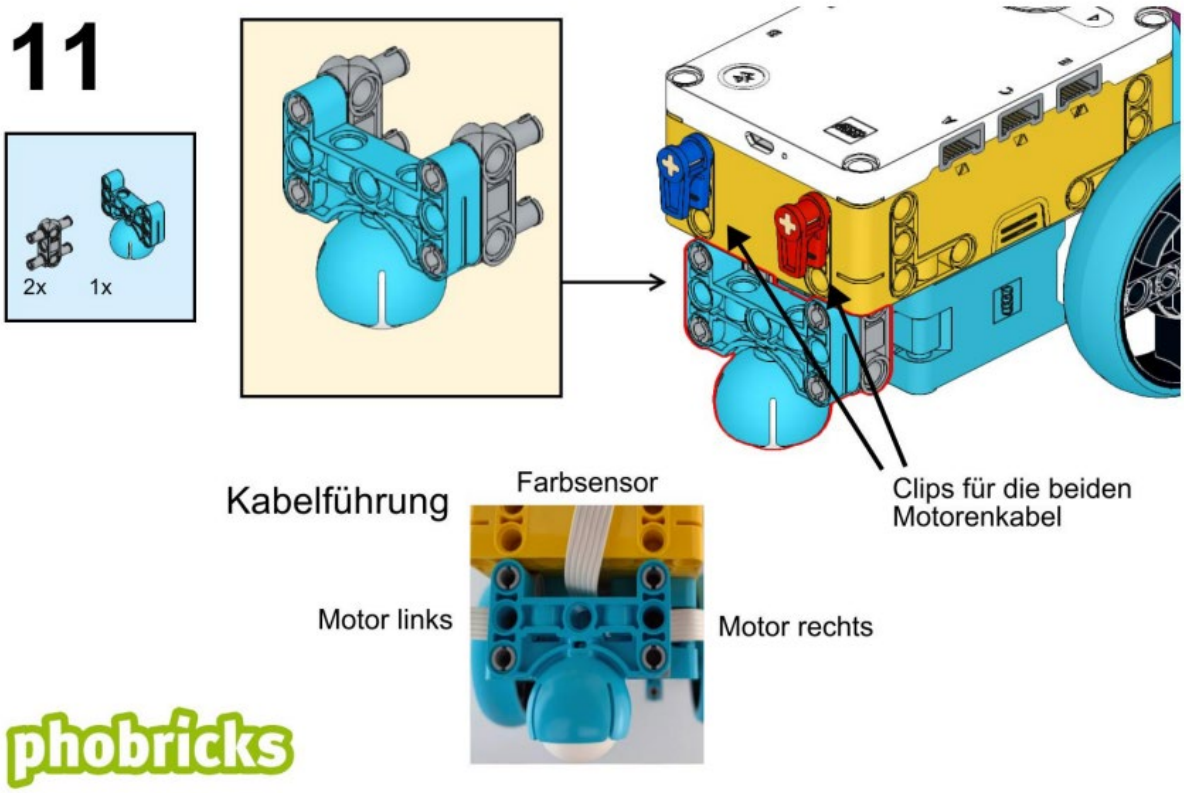




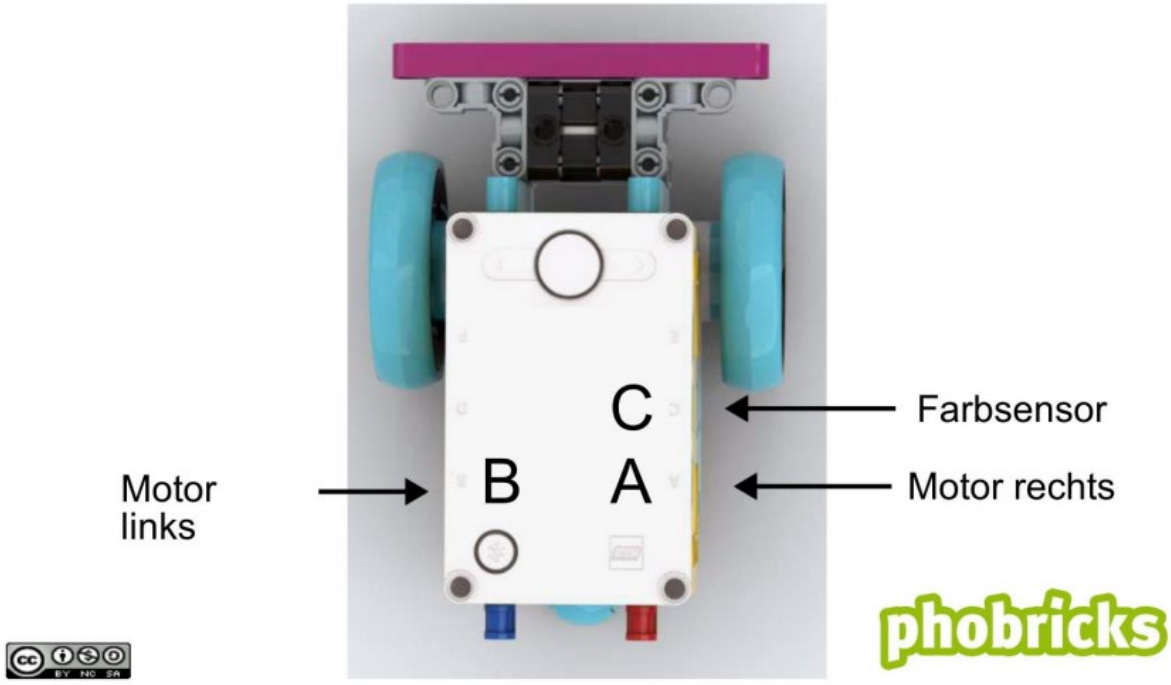








Anschlüsse



Didaktischer Hinweis

Dies ist ein sehr einfacher, schnell gebauter Roboter, der für die kommenden Aufgaben in diesem Dossier ausreicht. Für einen Wettbewerb muss er um Werkzeuge und weitere Sensoren erweitert werden!

7: Wir lassen unseren Roboter fahren

Bei den ersten Schritten hast du einen Motor bewegt. Wenn der Roboter aber wie ein Auto fahren soll, müssen wir zwei Motoren gleichzeitig bewegen.

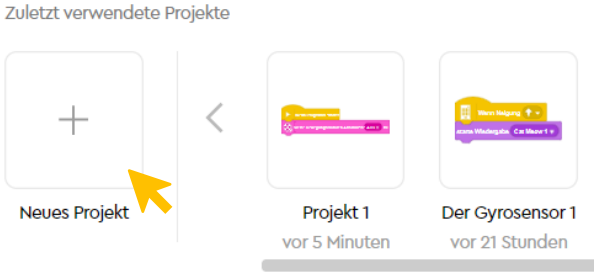
7.1 Fange ein neues Projekt an

Lernziele

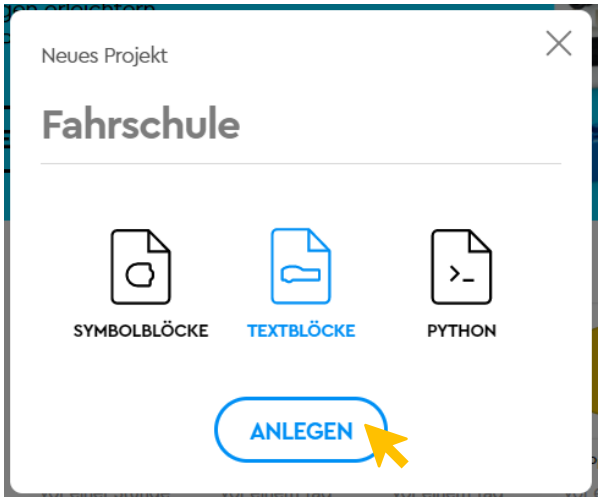
- ☐ Ich weiss, wie ich in der SPIKE App ein neues Projekt starte.

Gehe in der SPIKE App auf die **Startseite**. Du findest sie von überall in der App mit dem kleinen Haus-Symbol.

Klicke dann auf **Neues Projekt** (unter dem blauen Bereich mit den Ersten Schritten).



Jetzt siehst du ein kleines Fenster, in dem oben hellgrau «Projekt 1» steht. Schreibe dort einen guten Namen für dein Projekt hin, damit du es später leicht wieder findest, z.B. «Fahrschule». Dann wählst du **Textblöcke** aus und klickst auf **Anlegen**.



Didaktischer Kommentar

Es bietet sich an, darauf zu achten, dass die Schülerinnen und Schüler es sich angewöhnen, allen Projekten immer gleich einen sinnvollen Namen zu geben. Sich später noch daran zu erinnern, ob das Projekt von der letzten Stunde nun Projekt18 oder Projekt19 war, ist dann doch sehr mühsam. Projekte können jederzeit umbenannt oder auch wieder gelöscht werden. Und noch gut zu wissen: die SPIKE App speichert immer automatisch! (Dies ist andererseits ein Nachteil bei komplexeren Programmen, wo es sinnvoll ist, verschiedene Versionen zu speichern um im Falle eines Fehlers zur letzten funktionierenden Version zurückkehren zu können, aber wenn die Schülerinnen und Schüler sich angewöhnen, oft unter einem neuen Namen zu speichern, kann man auch dieses Problem umgehen. Bsp.: Originaler Projektname ist «Fahrschule», dann «Fahrschule 15.11. 14:03», dann «Fahrschule 15.11. 16:10»,(

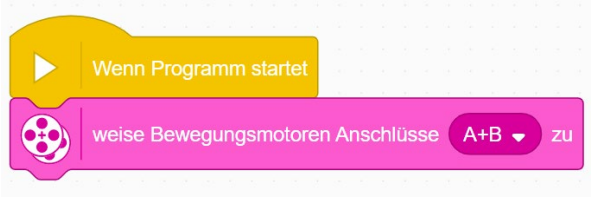
7.2 Antriebsmotoren festlegen

Lernziele

- ☐ Ich weiss, was Antriebsmotoren sind und wie ich sie für ein Projekt festlegen kann.

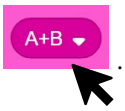
Als erstes müssen wir dem Programm beibringen, welche Motoren der Roboter zum Fahren braucht, und welcher davon rechts und links ist. Das nennt man die **Antriebsmotoren**.

Tipp:
Wenn wir die Antriebsmotoren gleich am Anfang festlegen, weiss das Programm immer Bescheid und wir müssen es nicht jedes Mal extra dazu sagen. Das ist vor allem bei längeren Programmen wichtig, wo immer wieder gefahren, angehalten, oder die Richtung geändert wird.



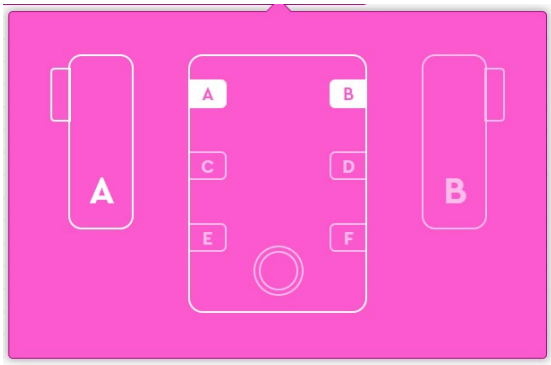
Dies ist die Standardeinstellung dieses Blocks, und sie ist perfekt, wenn dein **linker Moter** im Anschluss **A** eingesteckt ist, und dein **rechter Motor** im Anschluss **B**. Logischerweise müssen wir andere Buchstaben auswählen, wenn deine Motoren nicht in den Anschlüssen A und B eingesteckt sind.

Fall nach dem Start deines Programms der Roboter in die verkehrte Richtung fährt (also rückwärts statt vorwärts), musst du die Zuweisung der Bewegungsmotoren ändern. Natürlich könntest du einfach die Kabel aus- so wieder einstecken, dass A links und B rechts ist. Aber je nach Konstruktionsweise deines Roboters ist das vielleicht gar nicht möglich. Darum kannst du die Zuweisung auch im Code ändern. Das geht so:

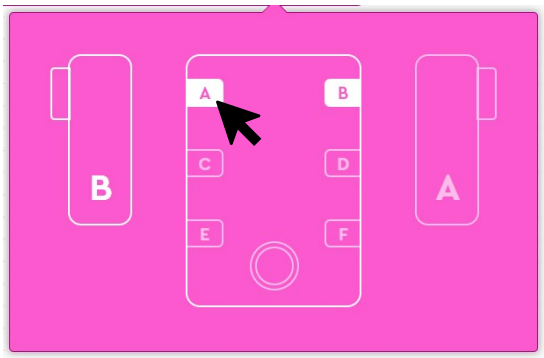
Klicke auf das Einstellungsfeld 

.

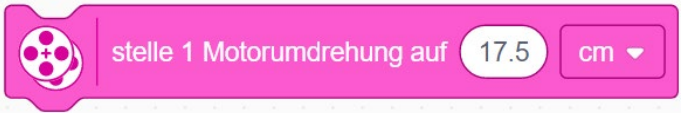
Du siehst nun dieses Fenster:



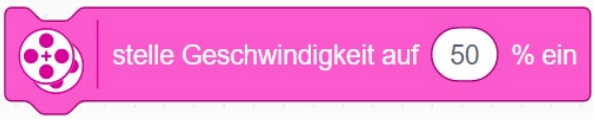
Du siehst darin, an welchen beiden Anschlüssen die Motoren stecken, und welcher Motor links ist. Wenn du jetzt auf das einen der beiden Anschlüsse klickst, wird die Zuordnung umgedreht: Jetzt ist **B links** und **A rechts**.



Wir können auch gleich festlegen, wie gross die Räder sind (Radumfang) sind, da es natürlich einen Unterschied macht, ob wir mit grossen oder kleinen Rädern fahren. Bei grösseren Rädern fährt der Roboter mit einer Umdrehung weiter als mit kleinen Rädern. Die türkisen Räder, die in der SPIKE-Prime-Box sind, sind 17.5 cm gross, was die Standard-Einstellung ist, falls du das mal vergisst. Aber es ist guter Stil, die Radgrösse trotzdem immer anzugeben.



Zuletzt stellen wir die Standardgeschwindigkeit ein. Sie wird immer dann verwendet, wenn wir nichts anderes angeben. Geschwindigkeiten werden immer in Prozent angegeben. 0% bedeutet keine Bewegung, und 100% ist die Höchstgeschwindigkeit des Motors. Wenn du eine Zahl schreibst, die grösser als 100 ist, wird der Roboter das automatisch als 100 interpretieren. Du kannst den Roboter nicht zwingen, schneller zu fahren – ausser du verwendest Zahnräder, aber das ist ein anderes Thema und wird hier nicht behandelt.



Tipp:
Für Übungen am besten eine niedrige Geschwindigkeit wählen. Meistens ist der Roboter dann ein bisschen präziser. Man kann auch besser beobachten, was geschieht, und allfällige Fehler erkennen. Später kannst du immer noch schneller werden!

Am Ende der Zuweisung der Bewegungsmotoren, sieht dein Code so aus:

Diese drei Bewegungsblöcke müssen am Anfang jedes Programms stehen, in dem dein Roboter irgendwohin fahren soll. Natürlich musst du jeweils die richtigen Motoren auswählen und den Radumfang sowie die Geschwindigkeit. richtig einstellen.

Didaktischer Kommentar

Die Antriebsmotoren am Anfang des Projekts festzulegen hat mehrere grosse Vorteile:

- *Entfernungen können in cm angegeben werden; man muss also nicht selbst rechnen, wie viele Umdrehungen oder Grad es braucht, bis eine bestimmte Strecke erreicht ist.*
- *Die Standardgeschwindigkeit gilt immer dann, wenn nichts anderes angegeben ist.*
- *Durch das Festlegen, welcher Motor links und welcher rechts ist, haben wir den Vorteil, dass wir immer «richtig herum denken» können. Es könnte für ein Projekt nötig sein, die Motoren anders anzubauen, so dass sie für uns rückwärtsfahren, obwohl sie laut Programm eigentlich vorwärtsfahren. In einer solchen Situation müsste man dann immer entweder die Entfernung oder die Geschwindigkeit mit -1 multiplizieren – eine häufige Fehlerquelle! Da unser Roboter aber weiss, welcher Motor links und welcher rechts ist, übernimmt er diese Denkarbeit für uns.*

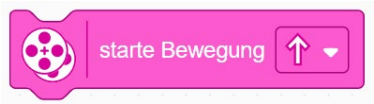
7.3 Einfaches Fahren

Lernziele

- ☐ Ich kann meinen Roboter starten und mit einem «Warten»-Block zum Anhalten bringen.

Jetzt sind wir endlich bereit, unseren Roboter fahren zu lassen!

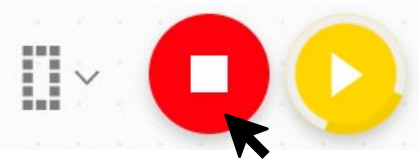
Unter der Festlegung der Antriebmotoren kommt ein **«Starten»-Block**: Der Pfeil nach oben zeigt an, dass der Roboter vorwärts fahren soll.



«Starten»-Block

Was passiert? Wie bekommst du jetzt den Roboter wieder zum Anhalten? Du willst ja nicht, dass er so lange fährt, bis der Akku leer ist!

Drücke drum zum Anhalten auf den runden Knopf am Roboter, oder klicke den roten Stopp-Knopf in der SPIKE-App:



Aber wir wollen, dass der Roboter von selbst anhält!

Jetzt wollen wir, dass der Roboter genau 1 Sekunde lang fährt und dann wieder anhält. Dafür brauchen wir einen orangenen **Warten-Block** und dann einen **Anhalten-Block**:

«Starte Bewegung»-Block

«Warten»-Block

«Anhalten»-Block

7.4 Anhalten an der schwarzen Linie mit einem «Warte-bis»-Block

Lernziele

- ☐ Ich kenne den «Wenn-dann»-Block und kann ihn verwenden, um meinen Roboter zu stoppen.
- ☐ Ich kann den Farbsensor verwenden.

Jetzt wollen wir nicht eine Sekunde lang fahren, sondern genau so lange, bis wir an einer schwarzen Linie ankommen.

Wir brauchen jetzt statt dem einfachen Warte-Block einen **Warten-bis-Block**:

«Warten-bis»-Block

In das Sechseck muss die Bedingung «an einer schwarzen Linie ankommen» verpackt werden. Dafür brauchen wir einen **Farbsensor**. Wir gehen also zu den hellblauen Sensor-Blöcken und suchen einen aus, mit dem die Farbe gemessen wird. Achte auch darauf, dass du beim Farbsensor den richtigen Anschluss (**C**) auswählst!

Tipp:
Für alle Sensoren gibt es ovale Blöcke, die dir immer den aktuellen Messwert angeben, und die sechseckigen Blöcke, die den aktuellen Messwert mit einem gewünschten Wert vergleichen. Im «Warten bis»-Block brauchen wir immer den sechseckigen Block!

Wir brauchen den türkisen «**Farbe?**»-Block:

Aber aufgepasst! Mit dieser Einstellung sucht der Sensor nicht nach schwarz, wie es in der Aufgabe steht, sondern nach rot! Klicke darum auf den kleinen Pfeil im Farbfeld und wähle schwarz aus. Füge dann diesen Block in das sechseckige Feld im «**Warte bis**»-Block.

Der ganze Code (nach der Festlegung der Antriebsmotoren) sieht dann so aus:

«Farbe-gleich?»-Block in einem «Warten-bis»-Block

Aufgabe:

Dein Roboter soll nicht nur anhalten, sondern auch in der Lichtmatrix ein «X» anzeigen und einen Piepton abspielen. Wie musst du den Code ergänzen?

Didaktischer Kommentar

Zum Fahren gehört immer eine Richtung. Die wird im «starten»- Block ausgewählt. Aber man muss auch wieder anhalten. «Warten»-Blöcke sind eine einfache Methode festzulegen, wann angehalten werden soll.

Weiterführende Ideen:

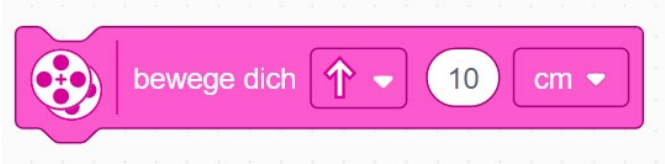
- *Es bietet sich an, im «warten-bis»-Block auch andere Bedingungen für das Anhalten festzulegen. Dafür kann man auch andere Sensoren an den Roboter anbauen.*

7.5 Der Block «Für eine bestimmte Dauer bewegen»

Lernziele

- ☐ Ich kenne den Block «für eine bestimmte Dauer bewegen».

Der Starten--Block ist dann nützlich, wenn wir nicht genau wissen, wie weit der Roboter fahren muss. Oft wissen wir es aber, weil wir ein Lineal haben und die Strecke genau kennen. Dann können wir auch den **Block «für eine bestimmte Dauer bewegen»** nehmen.



Aufgaben:

1. Was musst du tun, um den Roboter rückwärtsfahren zu lassen?
2. Lass deinen Roboter genau 29 cm weit vorwärtsfahren. Überprüfe mit einem A4-Blatt, ob es stimmt!
3. Lass deinen Roboter genau eine Umdrehung weit vorwärtsfahren. Wie viele Zentimeter sind das?
4. Lass deinen Roboter mit Geschwindigkeit 20% genau 3 Sekunden lang neben einem Lineal vorwärtsfahren. Ändere dann bei den Antriebsmotoren die Standard-Geschwindigkeit auf 40% und lass ihn nochmal 3 Sekunden lang vorwärtsfahren. Was beobachtest du?
5. Lass deinen Roboter eine Kreiseldrehung machen. Wie viele cm musst du eingeben, bis er wieder am Ende genauso steht wie am Anfang?

Didaktischer Kommentar

Wenn die Länge der zu fahrenden Strecke bekannt ist, dann kann man mit diesem Block bereits die Stopp-Bedingung festlegen: Nämlich genau dann, wenn die Länge der Strecke erreicht ist. Technisch gesehen, rechnet der Roboter die Zentimeter in Umdrehungen bzw. Grad (1/360 Umdrehung) um, und sobald diese Zahl überschritten ist, wird gestoppt. Auch hier kann es, insbesondere bei hohen Geschwindigkeiten zu kleinen Messfehlern kommen. Wenn man feststellt, dass aus 10 cm etwa systematisch 10.2 cm werden, und aus 23 cm ebenso systematisch 23.2, dann kann man das durch die Eingabe eines kleineren Wertes ausgleichen.

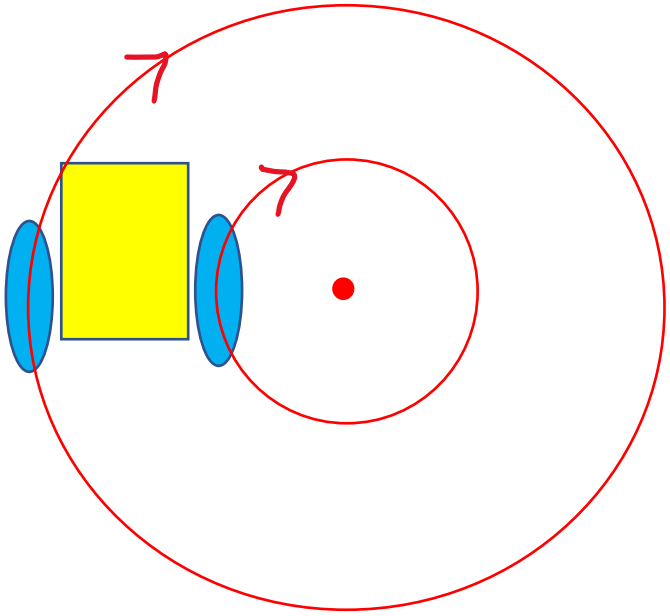
7.6 Drehen und Abbiegen

Lernziele

- ☐ Ich kenne verschiedene Möglichkeiten, meinen Roboter abbiegen zu lassen.

Bis jetzt kann dein Roboter geradeaus fahren, und du hast auch mal ausprobiert, ihn rückwärtsfahren zu lassen. Er kann sich aber auch drehen und abbiegen. Dafür gibt es mehrere Möglichkeiten:

1. **Kurve**: Du kannst eine Kurve fahren (die eigentlich ein Teil eines Kreises ist:

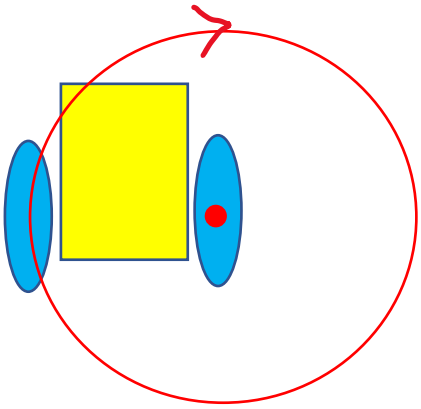


Dafür brauchen wir den **Block «Lenken für eine begrenzte Dauer»**:

Aufgabe:

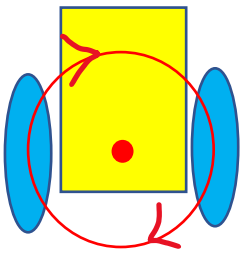
Probiere aus, was passiert, wenn du die Zahl im ersten Einstellungsfeld grösser oder kleiner machst!

2. **Zirkeldrehung**: Eine spezielle Form der Kurve ist die «Zirkeldrehung». Sie funktioniert genau wie ein Zirkel in der Geometrie: Ein Rad bleibt an der Stelle, das andere fährt.



Für diese Art von Drehung brauchen wir immer noch denselben Block **«Lenken für eine begrenzte Dauer»**. Aber jetzt müssen wir den Wert im Feld Links/Rechts auf genau **50** setzen, wenn wir uns nach rechts drehen wollen, und auf **-50**, wenn wir uns nach links drehen wollen.

3. **Kreiseldrehung**: Bei einer Kreiseldrehung dreht der Roboter sich um die Mitte zwischen seinen beiden Antriebsrädern.



Diese Drehung ist am einfachsten: Wähle einfach den **gebogenen Pfeil** im **Block «für eine bestimmte Dauer bewegen»**

Aufgabe:

Wie viele Zentimeter brauchst du für eine volle Kreiseldrehung?

Wenn du etwas Zeit zum Ausprobieren hast, dann baue doch deinen Roboter um, damit er breiter wird. Finde jetzt wieder heraus, wie viele Zentimeter du für eine Kreiseldrehung brauchst!

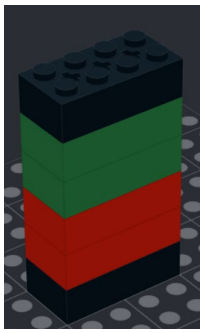
7.7 Bedingungen

Lernziele

- ☐ Ich kann meinen Roboter so umbauen, dass er für eine andere Aufgabe geeignet ist.
- ☐ Ich weiss, wie ich wenn-dann-Bedingungen und Wenn-dann-sonst-Bedingungen programmiere.

Jetzt wollen wir noch einmal zu unserem Selbstfahrenden Auto-Roboter zurückkommen. Erinnerst du dich noch? Er muss an der gelben Linie überprüfen, welche Farbe die Ampel hat, und sich dann entsprechend verhalten. Wir wollen genau das jetzt programmieren.

Leider gibt es in der SPIKE Prime-Box nur einen Farbsenspr, also müssen wir den verwenden, um die Ampel zu erkennen. Der Roboter muss also so umgebaut werden, dass er nach vorne schaut statt nach unten. Sei kreativ, es ist nicht schwierig! Danach muss du noch ein oder zwei Objekte bauen die du als Ampel verwenden kannst. Die Farbe muss dabei auf derselben Höhe sein wie dein Farbsensor! Dieses Bild kann dir als Hinweis helfen, aber je nachdem, wie du den Sensor angebaut hast, sieht dein Objekt vielleicht ganz anders aus.



Stelle jetzt den Roboter auf deinen Tisch, ohne dass irgendetwas vor dem Farbsensor ist.

Kontrolliere, ob dein Hub eingeschaltet und verbunden ist.

Kontrolliere jetzt, was dein Sensor sieht:

Du findest diese kleine Infografik oben links in deinem Projekt. Du siehst dort, an welchen Anschlüssen deine Motoren eingesteckt sind (**A** und **B**), und in welcher Position sie gerade sind. Wir interessieren uns jetzt aber dafür, was uns der Farbsensor im Anschluss **C** sagt: Er erkennt gerade keine Farbe. Das bedeutet dieses Zeichen: 

Stelle jetzt deine «Ampel» vor den Sensor, so dass ungefähr ein Finger breit Platz hat. Jetzt sollte dir der Farbsensor im Anschluss **C** die richtige Farbe zeigen.

Wir starten unseren Roboter wie immer mit einem «**Starte Bewegung**»-Block und einem «**Warte bis**»-Block.

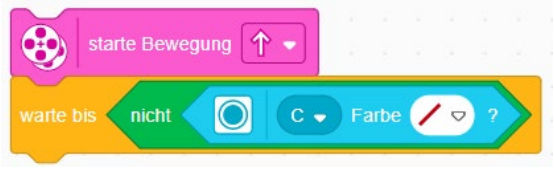
Wir brauchen jetzt die Unterscheidung zwischen keiner Farbe und entweder Grün oder Rot.

Zuerst wollen wir überprüfen, ob überhaupt eine Farbe erkannt wird. Dazu benützen wir den «**Farbe?**»-Block:

Aber wir brauchen ja eine Farbe! Einen Block für «Sieht der Sensor eine beliebige Farbe?» Gibt es leider nicht. Darum brauchen wir einen Block, der diesen Sensorblock in sein Gegenteil umdreht. Das ist der «**Nicht**»-Block.

Wie du siehst, hat dieser sechseckige Block Platz für einen weiteren sechseckigen Block, also ziehen wir einfach unseren türkisen «**Farbe?**»-Block in den grünen «**Nicht**»-Block hinein.

Und dann schieben wir beides zusammen in einen «**Warte bis**»-Block.



Wir fahren damit, bis der Sensor das Ampel-Objekt entdeckt.

Jetzt soll er sich entscheiden: Anhalten oder weiterfahren?

Für eine Entscheidung brauchen wir wieder einen C-Block, den «Wenn-dann»-Block.

Überlege jetzt: Was ist wichtiger: Dass der Roboter Rot entdeckt, oder dass er Grün entdeckt? Im Strassenverkehr ist es nicht so schlimm, falls du die grüne Ampel übersiehst. Du darfst schliesslich einfach weiterfahren. Aber wenn du die rote Ampel übersiehst, kann das gefährlich werden. Also suchen wir nach der roten Ampel.

Merke:

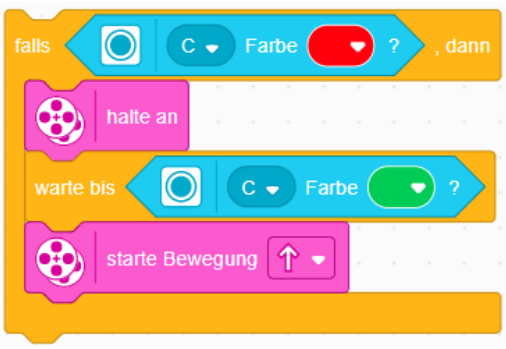
Suche immer zuerst nach dem, was am wichtigsten ist!

Dashalb fügen wir jetzt den «**Farbe?**»-Block mit der gesuchten Farbe in das Sechseck des «**Wenn-Dann**»-Blocks.

Und im C-Block schreiben wir, was wir tun sollen, wenn diese Bedingung erfüllt ist:

Mit diesem Code hält der Roboter nur an, wenn die entdeckte Farbe rot ist. Wenn sie nicht rot ist, macht der Roboter mit dem weiter, was unterhalb des C-Blocks steht.

Wir können den Code so ändern, dass der Roboter dann weiterfährt, wenn er grün sieht:



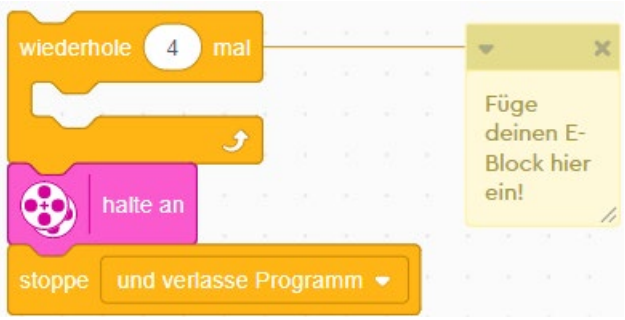
Wenn der Roboter noch etwas spezielles machen soll, wenn er grün entdeckt, brauchen wir statt dem C-förmigen «**Wenn-Dann**»-Block den E-förmigen «**Wenn-Dann-Sonst**»-Block.

In unserem Beispiel soll der Roboter kurz piepsen, damit wir sehen, dass er die grüne Ampel richtig erkannt hat – denn er fährt ja bereits, also brauchen wir keinen neuen «**Starte Bewegung**»-Block!

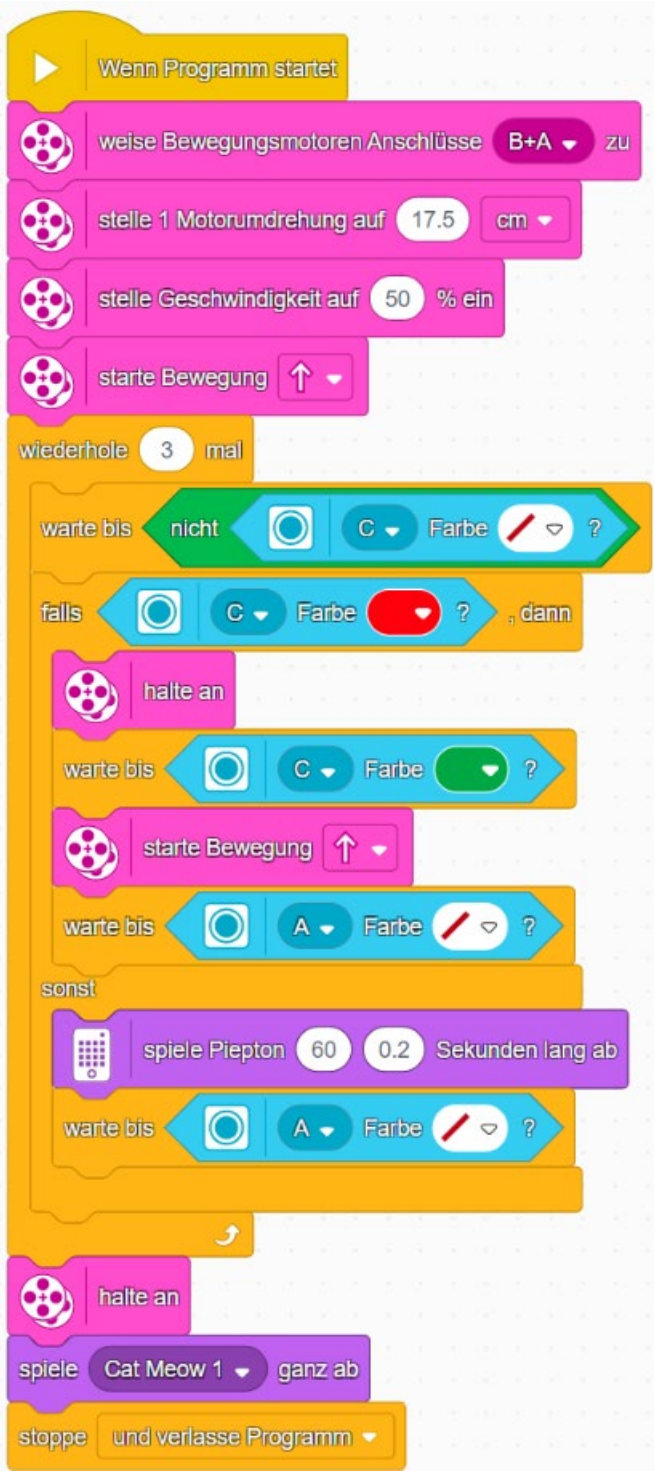
Unser Code sieht jetzt so aus:

Die beiden Blöcke, mit denen gewartet wird, bis keine Farbe mehr zu sehen ist, sind ein Trick, mit dem du Zeit gewinnst, deine Ampel-Blöcke umzustellen.

Wenn wir wissen, dass es auf unserer Strecke 4 Ampeln hintereinander gibt, können wir noch eine Schleife um den E-Block herum machen. Und am Ende der Schleife musst du dann noch anhalten und das Programm beenden, sonst läuft dein Roboter unendlich weiter!



Der komplette Code sieht jetzt so aus:



Didaktischer Kommentar

In diesem Abschnitt geht es ums Abbiegen und Drehen. Das geht mit unterschiedlich grossen Wendekreisen. Die Zahl im Einstellungsfehlt «nach rechts» bzw. «nach links» geben dabei an, wie eng die Kurve ist, die gefahren wird. 0 bedeutet, der Roboter bewegt sich geradeaus. 100 bzw. -100 bedeutet, dass der Roboter sich um seine eigene Achse dreht (Kreiseldrehung). Es dreht sich also ein Rad nach vorne, das andere genau gleich schnell nach hinten. Bei «nach rechts: 50» bleibt das rechte Rad auf der Stelle, während das linke vorwärtsfährt, und bei «nach

links: -50» ist es umgekehrt. Bei allen anderen Geschwindigkeiten drehen sich die Räder ungleich schnell, um die Drehbewegung zu erzeugen. Je nach Aufgabe und Platz auf dem Parcours kann die eine oder andere Art des Abbiegens sinnvoller sein.

Teil 4: Wichtige Algorithmen

Merke:

Ein Algorithmus ist eine Aufgabe, die in einem Programm immer wieder vorkommt und selbständig funktioniert.

8: «Bleibe in der Arena»

Lernziele

- ☐ Ich kenne den Begriff «Algorithmus»
- ☐ Ich kann das «Arena-Spiel» programmieren
- ☐ Für Fortgeschrittene: Ich kann das «Arena-Spiel» um weitere Ereignisse ergänzen

Bei dieser Aufgabe kombinierst du alles, was du bisher gelernt hast: Licht, Klänge, Bewegungen, Schleifen und Bedingungen. Schleifen und Bedingungen können auch ineinander verschachtelt werden!

Du brauchst hierfür auf dem Tisch oder dem Boden eine Fläche, die einen Rand aus schwarzem Klebeband hat. Diese sollte mindestens die Grösse A3 haben (grösser ist besser!). Sie kann jede beliebige Form haben, rechteckig, rund, wie eine Wolke…

Aufgabe:
Mache ein neues Projekt mit dem Namen «Arena-Spiel».

Dein Roboter soll immer innerhalb der schwarzen Linie bleiben. Wenn er an die Linie kommt, soll er:

- Stehen bleiben
- Kurz piepsen
- Auf der Lichtmatrix für 0.2 Sekunden einen Pfeil nach unten anzeigen
- In einem grossen Bogen rückwärts fahren (Einstellung: nach rechts: 30 / -150 Grad)
- Wieder vorwärts weiterfahren.

Das Ganze soll 10-mal wiederholt werden. Nach dem zehnten Mal soll der Roboter stehen bleiben und klatschen.

Tipp:
Versuche zuerst, die Aufgabe selbst zu lösen! Probiere immer wieder aus, ob alles so stimmt, wie du es willst, und verändere deinen Code entsprechend. Wenn du Schwierigkeiten hast, dann hast du diese Möglichkeiten:

- Versuche zunächst das zu programmieren, was der Roboter tun soll, wenn er an die schwarze Linie kommt. Kümmere dich erst danach darum, das in eine Bedingung zu verpacken.
- Wenn du mit der Struktur des Codes Schwierigkeiten hast, dann schau auf Seite 35 am Ende dieses Dossiers. Dort findest du einmal den Code, aber ohne Beschriftungen! Anhand der Farben und Formen kannst du überlegen, wie es gehen könnte. Und du findest auf Seite 36 den kompletten Code.
- Wenn das auch nicht weiterhilft, dann bitte die Kursleitung / deine Lehrperson um Hilfe.

9: Folge der Linie!

Lernziele

- ☐ Ich weiss, wofür ein Linienverfolger nützlich ist.
- ☐ Ich kann einen einfachen Linienverfolger programmieren.

Manchmal weisst du vorher nicht, welchen Weg dein Roboter genau nehmen muss. Oder der Weg ist so kompliziert, dass du ihn nicht einfach mithilfe der verschiedenen Bewegungs-Blöcke programmieren kannst. Dann gibt es oft eine sehr nützliche Hilfe: Schwarze Linien, denen du folgen kannst!

Aufgabe:

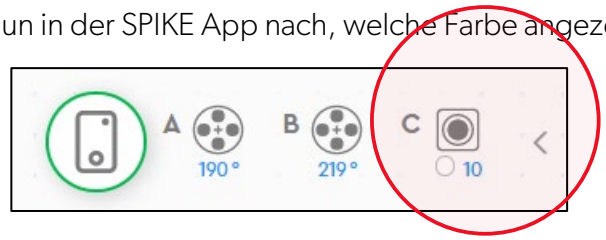
Auf dem Boden im Flur ist eine lange Linie aus Klebeband.
Du bist mit einem Partner/einer Partnerin ein Team.
Ein Teammitglied ist der «Roboter» und steht mit verbundenen Augen am Anfang der Linie.
Das andere Teammitglied ist der «Programmierer»/die «Programmiererin» und gibt ihm Anweisungen, wie es der Linie folgen soll.
Wechselt danach die Rollen.

Beobachte, wie der «Roboter» sich der Linie entlang bewegt!

Jetzt wollen wir einen Linienverfolger programmieren, der ganz ähnlich funktioniert.

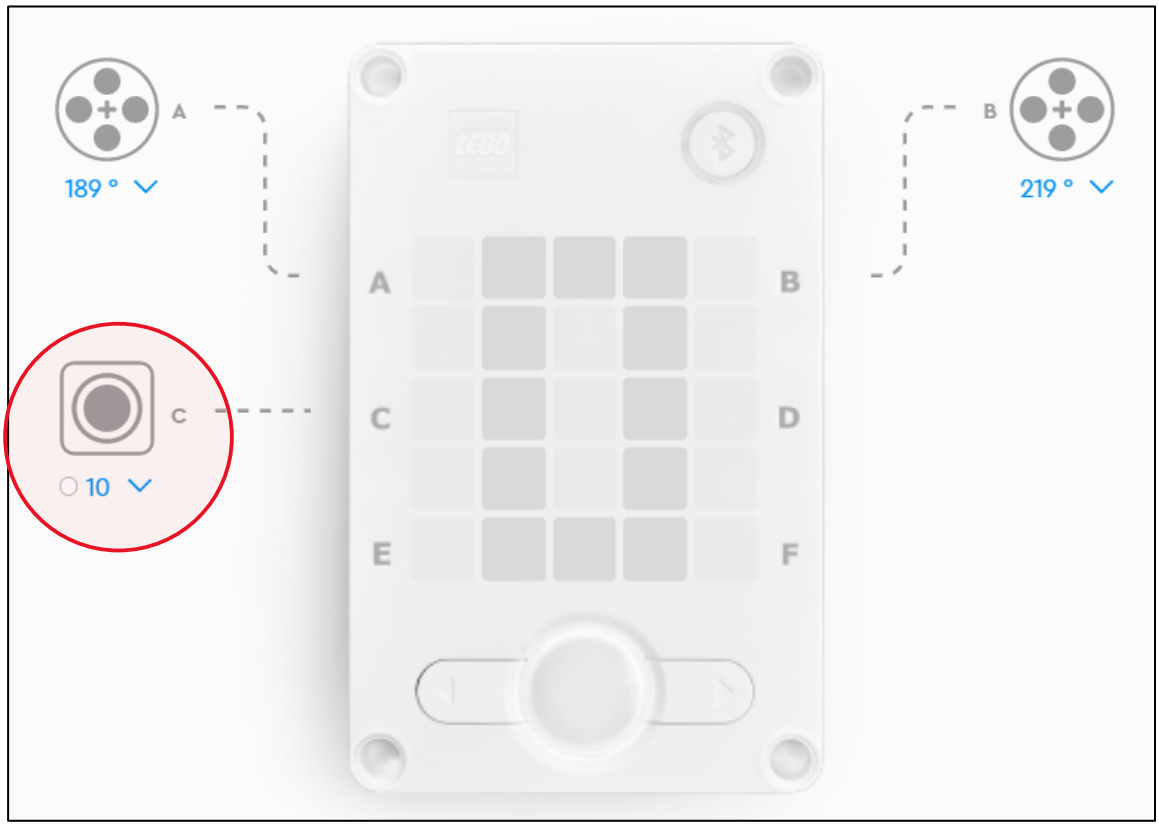
Vorher müssen wir aber noch etwas ausprobieren Du brauchst dafür wieder den **Farbsensor** vorne am Roboter.

Stelle deinen Roboter mit dem Farbsensor über eine weisse Fläche, z.B. ein Blatt Papier.
Schaue nun in der SPIKE App nach, welche Farbe angezeigt wird. Das sollte jetzt weiss sein.



Der Sensor zeigt Weiss an. «10» ist in SPIKE die Kennzahl für Weiss.

Klicke jetzt auf das Symbol für den Hub. Du kommst zu einer grösseren Ansicht des Hubs mit den angeschlossenen Motoren und Sensoren:



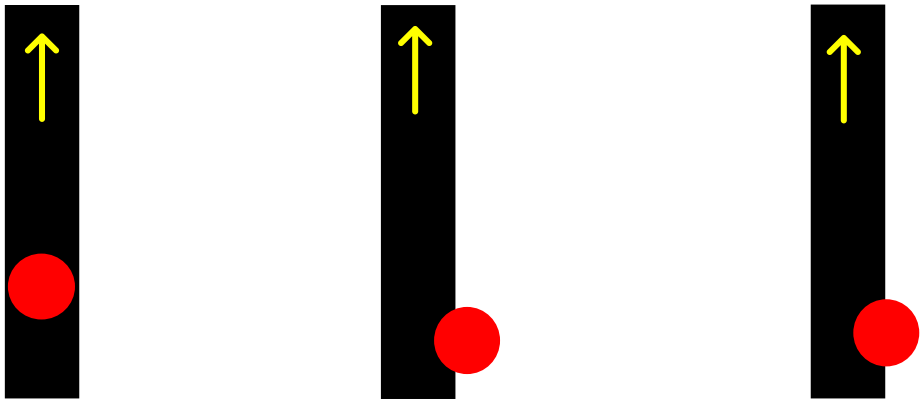
Wenn du auf den kleinen Pfeil unter dem Farbsensor-Symbol klickst, kommst zu einem Auswahlmenü. Wähle dort «Reflektieren»:

Wenn du die grosse Hub-Ansicht jetzt schliesst, sieht die Kontroll-Anzeige jetzt so aus:

Der Wert ist sehr hoch, fast 100%. Wenn es weniger liegt, kann das an einer schlechten Beleuchtung liegen, oder deine Unterlage ist nicht ganz weiss. Oder dein Sensor ist zu weit oben angebaut. Das ist nicht schlimm.

Wiederhole jetzt dasselbe für Schwarz. Der Wert ist sehr klein, meistens unter 20%.

Stelle dann deinen Roboter so hin, dass der Farbsensor genau über der rechten Kante (in Fahrtrichtung) der Linie ist, und miss den Sensorwert. Er sollte bei zwischen 40% und 60% liegen. Das ist unser **Schwellwert**.



✗ ganz auf der schwarzen Linie. Sensorwert zu tief.

✗ zu weit im weissen Bereich. Sensorwert zu hoch.

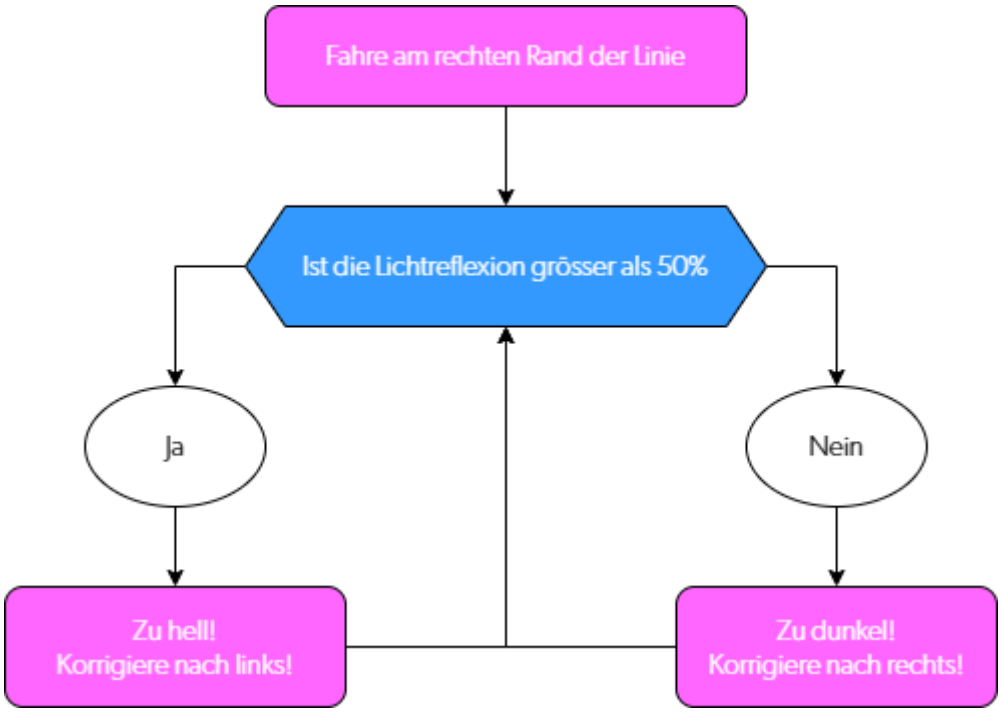
✓ genau richtig!

Erinnere dich jetzt an deine Übung an der schwarzen Linie: Wenn du nach rechts abgerutscht bist, musstest du nach links korrigieren, wenn du nach links abgerutscht bist, musstest du nach rechts korrigieren.

Genau dasselbe tun wir jetzt mit unserem Roboter. Wenn der Roboter nach rechts rutscht, wird der Farbsensor zu hell (also die Zahl wird zu gross), und er muss nach links korrigieren; wenn der Roboter nach Links abrutscht, wird der Sensorwert zu klein, also zu dunkel, und er muss nach rechts korrigieren.

Und das Ganze soll er unendlich lange wiederholen (oder so lange, bis wir das Programm beenden).

Für diese Übung wollen wir der Linie so lange folgen, bis der Roboter weniger als 15 cm von einem Hindernis entfernt ist. Deshalb müssen wir noch einen **Abstandssensor** vorne am Roboter anbauen.



So sieht der Code aus:

Wenn-dann-sonst-Block:

Bedingung: Lichtreflexion kleiner als 50% (oder kleiner als der Wert, den du auf der Kante gemessen hast)

Dann (wahr): Zu dunkel → Korrektur nach rechts

Sonst (falsch): Zu hell → Korrektur nach links

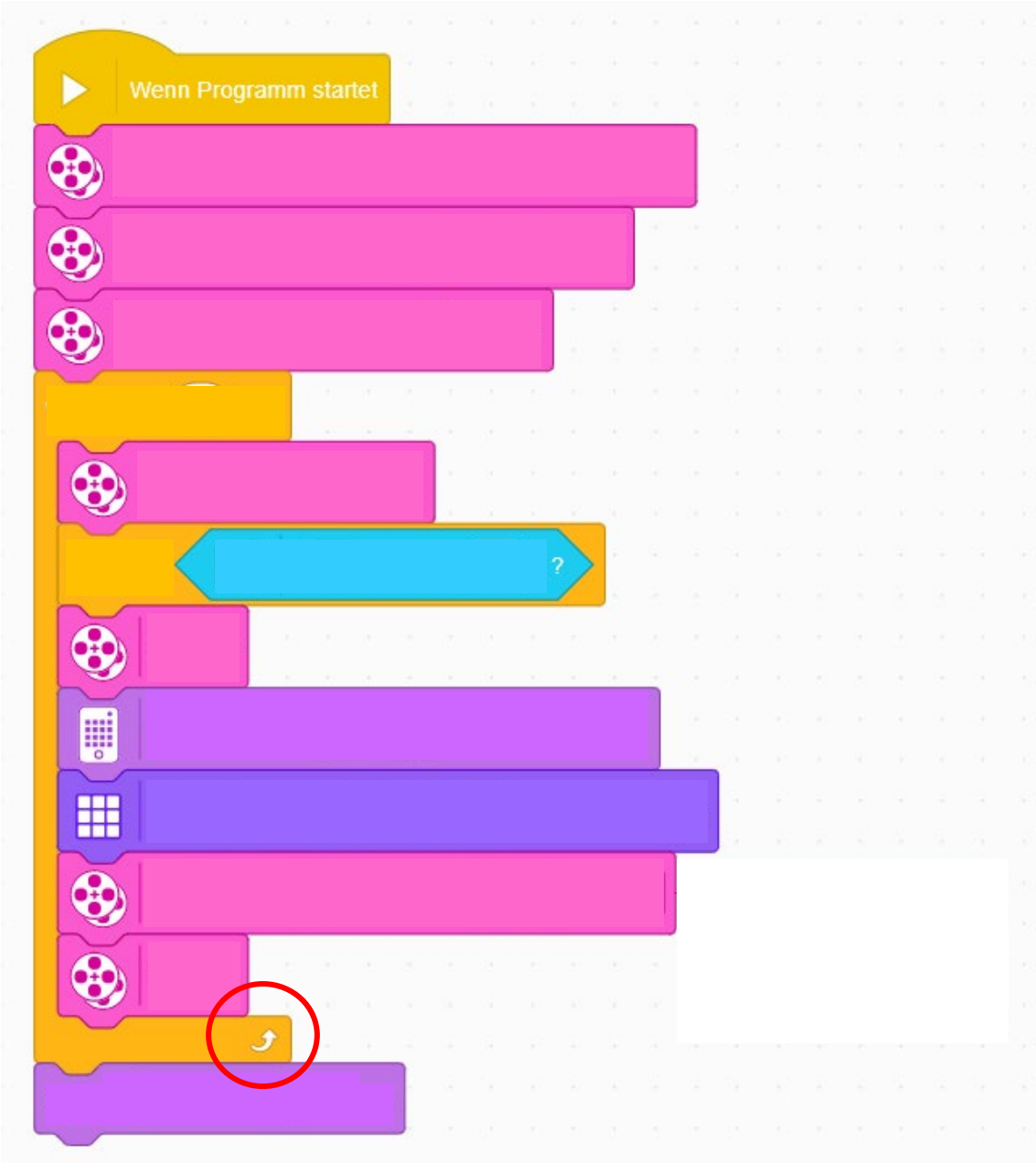
Beobachte: Wie fährt dein Roboter jetzt? Wie könnte man diese Art von Linienverfolger nennen?

Aufgabe:

Probiere aus, was passiert, wenn du einen grösseren oder kleineren Korrekturwert verwendest. Probiere auch aus, ob es bei den verschiedenen Korrekturwerten einen Unterschied macht, wie schnell dein Roboter funktioniert.

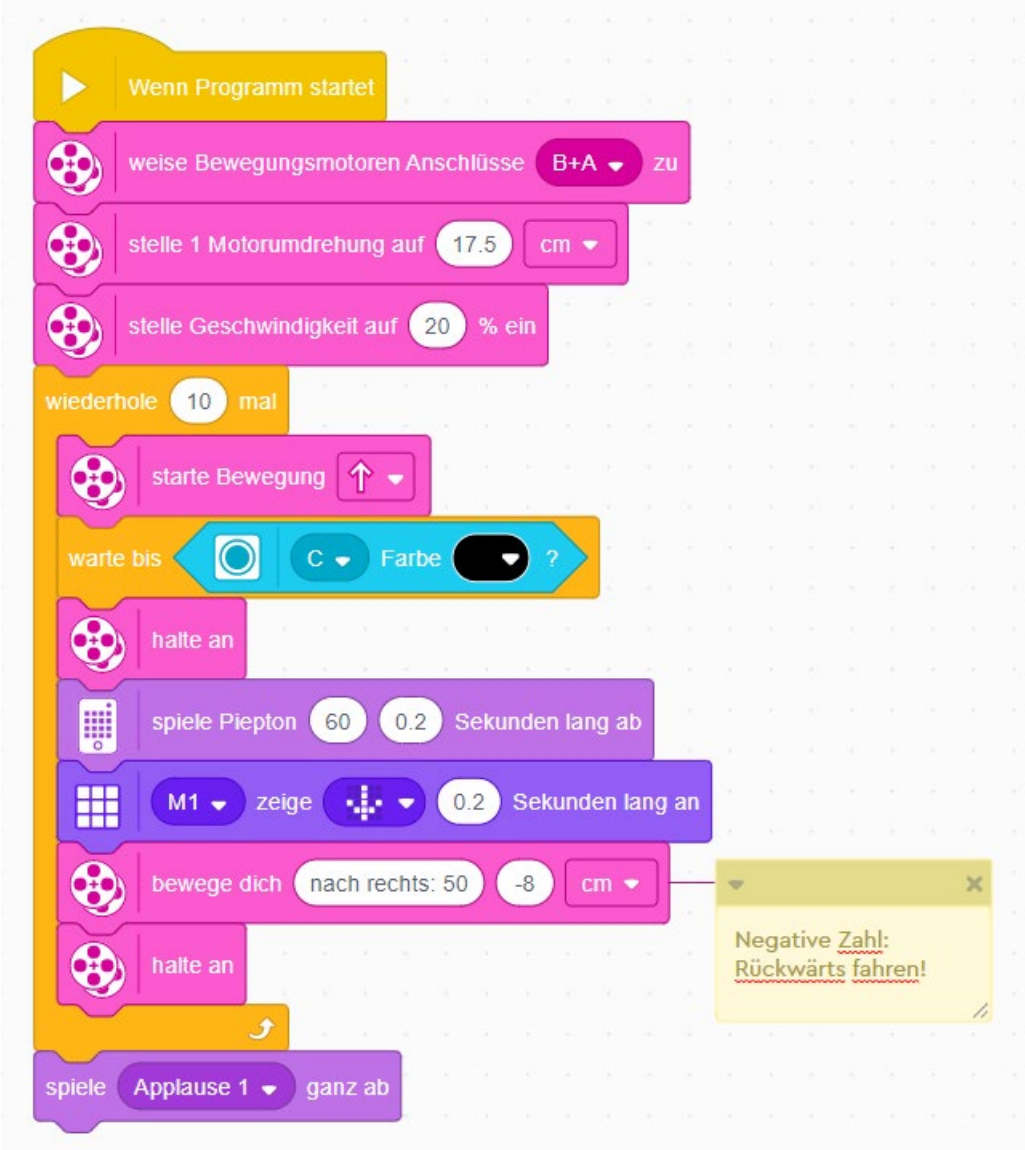
Hilfe zum Lösen der Aufgabe «Bleib in der Arena»

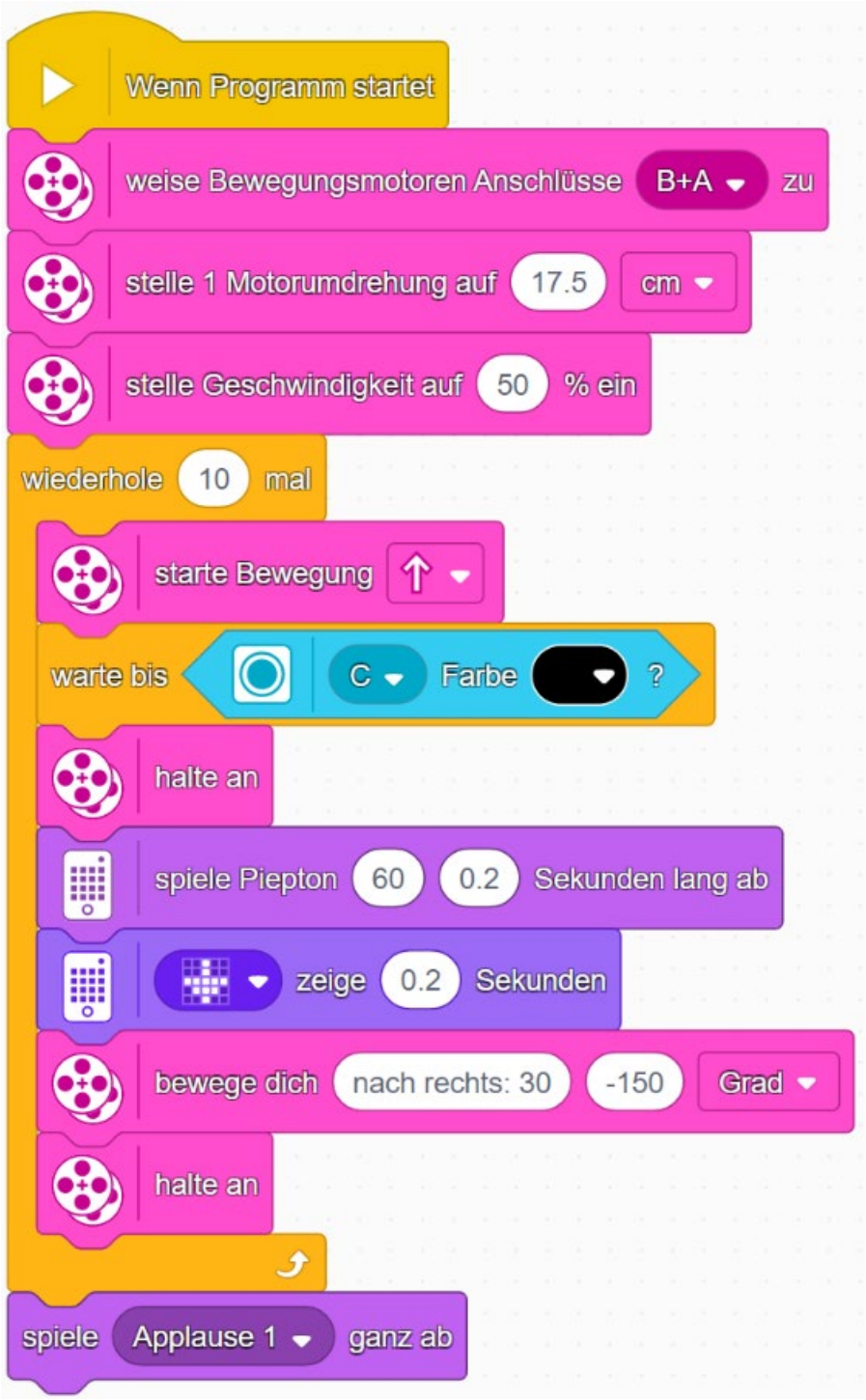
Denke daran: Am Anfang des Codes musst du die Antriebsmotoren festlegen. Im Warten-bis-Block legst du fest, was er tun soll, wenn er an eine schwarze Linie kommt, die er mit seinem Farbsensor erkennt.



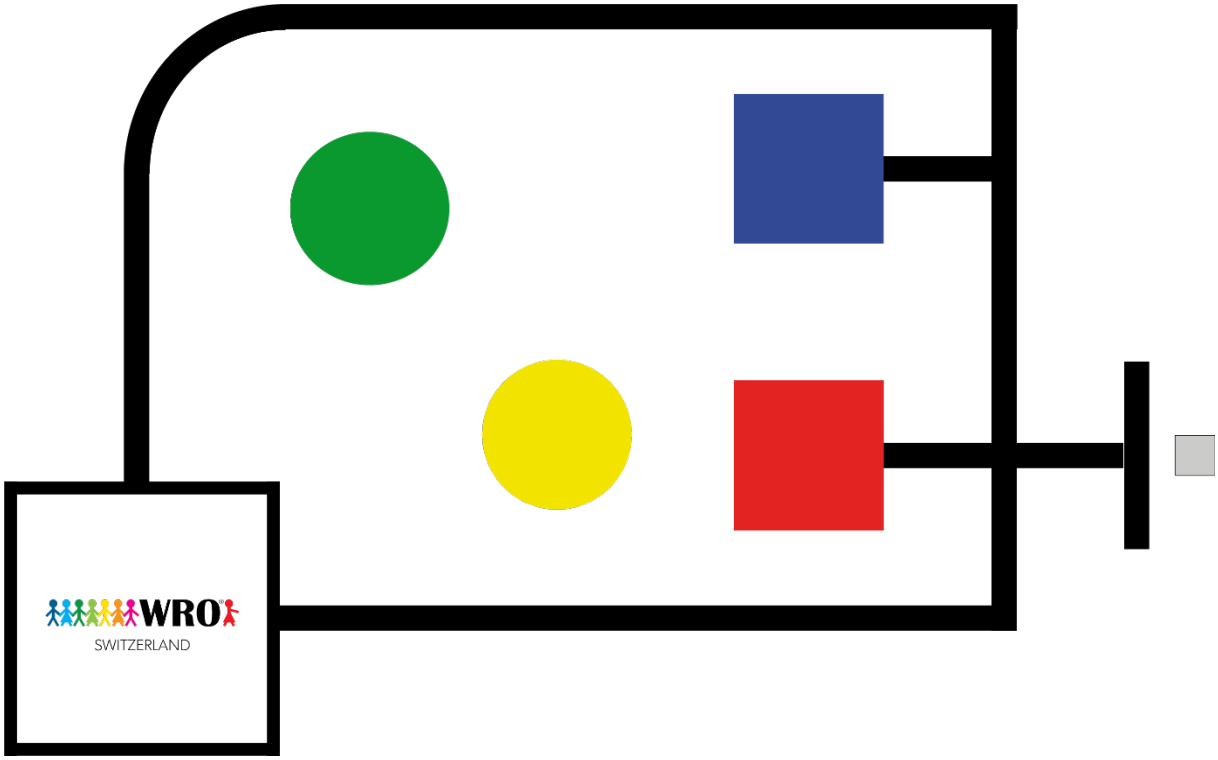
Lösung der Aufgabe «Bleib in der Arena»

Lösen der Aufgabe «Bleib in der Arena»Lösung der «Aufgabe bleib in der Arena»





Ideen zur Verwendung der Übungsmatte



- *Der Linie folgen*
- *Der Linie folgen bis zur Querlinie oder Abzweigung. Bei der Abzweigung aufpassen, welcher Seite der Linie der Farbsensor folgt!*
- *«Bleib in der Arena»*
- *«Bleib in der Arena mit «Boxauto»-Wettbewerb: Welcher Roboter schafft es, den anderen von der Matte zu verdrängen?*
- *Übungen zur Farberkennung mit unterschiedlichen Aktionen auf jeder Farbe*
- *4x4x2 Lego-Objekt vom Standort (rosa) in eines der Zielfelder (rot oder blau) bringen. Später das Objekt so verändern, dass die Grundfläche immer noch 4x4 ist, aber das Objekt höher und breiter wird. So konstruieren, dass man die Farbe entweder von oben oder von vorne (von der schwarzen Linie aus) erkennen muss.*
- *Roboter wieder korrekt im Start-Zielfeldfeld parkieren. (Dieses Feld ist kleiner als bei einem WRO-Wettbewerb. Aber für die Übungen kann auch der Roboter kompakter sein!*