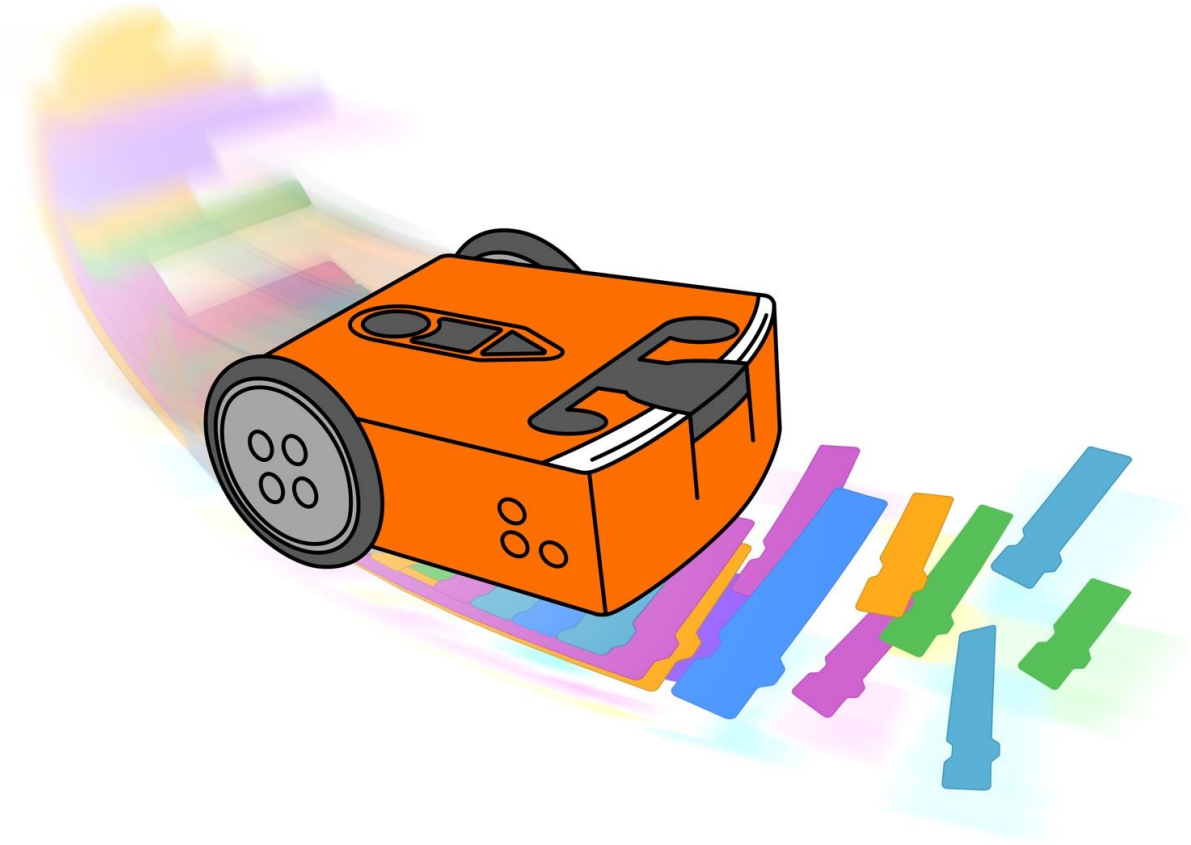
EdScratch-Unterrichtseinheiten

Arbeitsblätter und Aufgabenblätter für Schüler.



Die EdScratch-Lektionspläne, die von Kat Kennewell und Jin Peng erstellt wurden, stehen unter einer Creative Commons Attribution-ShareAlike 4.0 International License.

Inhalt

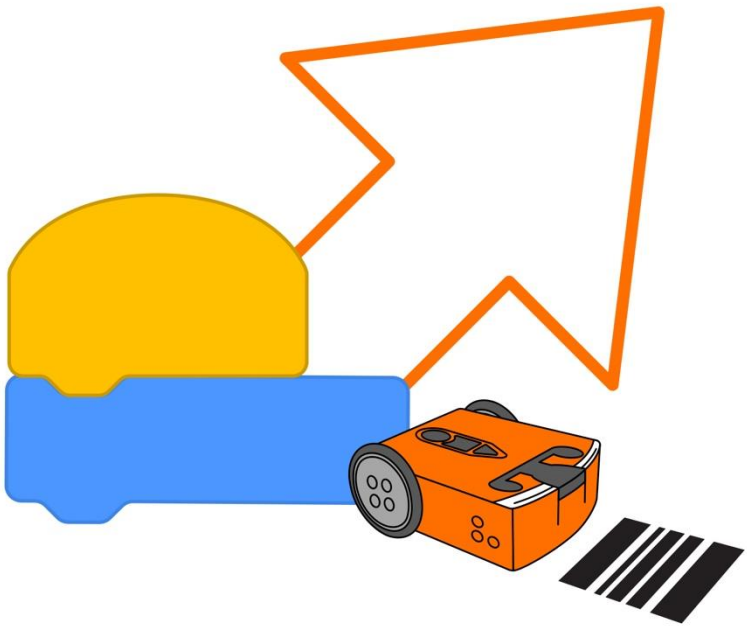
Einheit 1: Erste Schritte 6
U1-1.1: Dein Edison-Roboter 7
U1-1.1a: Bausteine, Blöcke und Edison 10
U1-1.2: Lass uns die Strichcode-Programmierung erforschen 11
U1-1.2a: Sumo-Ringen 18
U1-1.2b: Erstelle deinen eigenen Strichcode 20
U1-1.2c: TV-Fernbedienung Strichcodes 21
U1-1.2d: Edison-Fußball 24
U1-1.2e: Baue und kontrolliere den EdTank 25
U1-1.2f: Challenge: Baue und kontrolliere den EdDigger 26
U1-1.2g: Baue und kontrolliere die EdRoboClaw 27
U1-2.1: Lass uns die EdScratch-Umgebung erkunden 28
U1-2.1a: Lade ein weiteres Programm herunter! 34
U1-2.1b: Ist EdScratch = Scratch? 35
U1-2.2: Warnmeldungen 36
Arbeitsblatt U1-1: Grenze der Linienverfolgung 38
Arbeitsblatt U1-2: Sumo-Ring 39
Arbeitsblatt U1-3: TV-Fernbedienungen 40
Einheit 2: Bewegung! 41
U2-1.1: Lasst uns erforschen, wie Computer „denken“. 42
U2-1.1a: Mach ein Sandwich 46
U2-1.1b: Menschliche Roboter 48
U2-1.2: Gehen wir in EdScratch Schritt für Schritt vor 49
U2-1.3: Lass uns das Fahren mit Edison erforschen 51
U2-1.3a: Irrgarten-Wahnsinn 54
U2-1.3b: Selbstlaufendes Haustier 55
U2-2.1: Lass uns Edisons Ausgänge erforschen 56
U2-2.1a: Fahre sicher durch das Labyrinth 61
U2-2.2: Lass uns die Eingabeparameter untersuchen 63
U2-2.2a: Bring Edison bei, bis 9 zu zählen 66
U2-2.2b: Bring Edison bei, laut bis 9 zu zählen 69
U2-2.3: Lass uns Edisons musikalische Talente erkunden 71
U2-2.3a: Spiel ein Lied in einer Runde 77
U2-2.3b: Du bist der Dirigent 78
U2-2.4: Lass uns Bugs und Fehlerbehebung erforschen 79
U2-2.5: Lass uns Edisons Motoren erforschen 85
U2-2.5a: Drehender Garten 91
U2-2.5b: Sich drehendes Sonnensystem 92
U2-2.5c: Kartograph und Navigator 93

U2-2.5d: Autor und Regisseur 94
Arbeitsblatt U2-1: Gehe Schrit 94
Arbeitsblatt U2-2: Fahrstrecke 96
Arbeitsblatt U2-3: Mini- Labyrinth 97
Arbeitsblatt U2-4: Digitalanzeige 2 98
Arbeitsblatt U2-5: Digitalanzeige 5 99
Arbeitsblatt U2-6: Digitalanzeige 7 100
Arbeitsblatt U2-7: Digitalanzeige 8 101
Einheit 3: Schleifen programmieren 102
U3-1.1: sich wiederholenden Schritte 103
U3-1.1a: Fahre ein Dreieck 107
U3-1.1b: Fahre ein Sechseck 108
U3-1.1c: Wähle deine Form 109
U3-1.1d: Einen Kreis fahren 110
U3-1.1e: Ein Quadrat fahren 113
U3-1.1f: Doodle-bot-Herausforderung 114
U3-1.2: Schleifen und Sequenzen 115
U3-1.3: Unendliche Schleifen 117
U3-1.3a: Ohrwurm 120
U3-1.4: Das Stapeln und Verschachteln von Schleifen 121
U3-1.4a: Edison der Designer 127
U3-1.4b: Tanzparty! 128
U3-2.1: Das Unterbrechen des Hauptprogramms 129
U3-2.1a:Versuche es mit Klatschen 133
U3-2.1b: Betrüger-Bot 134
U3-2.1c: Wähle! 135
U3-2.2: Die Kommentarfunktion 136
U3-2.2a: Erstellen und kommentieren 140
U3-2.2b: Teile deine Kommentare 143
Arbeitsblatt U3-1: Ein Quadrat fahren 144
Arbeitsblatt U3-2: Ein Dreieck fahren 145
Arbeitsblatt U3-3: Ein Sechseck fahren 146
Arbeitsblatt U3-4: Einen Kreis fahren 147
Arbeitsblatt U3-5: Ein Viereck fahren 148
Arbeitsblatt U3-6: Quadrate wiederholen 149
Arbeitsblatt U3-7: Entwürfe zum Fahren 150
Einheit 4: Was wäre, wenn... 151
U4-1.1: Die Verwendung von Bedingten Anweisungen 152
U4-1.1a: Roboterfehler oder menschlicher Fehler? 158
U4-1.2: Die „Wenn - Dann“ - Funktion (if-statements) 164

U4-1.3: Wenn-Aussagen und Sequenzen 168
U4-1.4: Stapeln und Verschachteln von if-Anweisungen 172
U4-1.4a: Baue einen Flaschenzug 178
U4-2.1: Der Pseudocode 179
U4-2.1a: Finde die Antwort 183
U4-2.2: Werfen wir einen Blick auf Edisons Linienverfolger 185
U4-2.2a: Innerhalb einer Grenze fahren 189
U4-2.3: Betrachten wir Algorithmen näher 191
U4-2.3a: Es gibt mehr als einen Weg, einer Linie zu folgen 196
U4-2.4: Edisons Hinderniserkennung 199
U4-2.4a: Schneller, schneller, smash? 203
U4-2.4b: Wenn Linie, nach rechts gehen. Wenn Hindernis, gehe links 204
U4-2.4c: Wo ist das Hindernis? 206
U4-2.4d: 3D-Labyrinth 208
U4-2.5: Nachrichtenaustausch mit Edison 209
U4-2.5a: Ferngesteuerte Flaggenmaschine 214
U4-2.5b: Baue und steuere den EdCrane 217
U4-2.5c: Wasserwerfer zur Brandbekämpfung 220
U4-2.5d: Halbautomatischer Bagger 223
U4-2.5e: Entfernung von gefährlichem Material 226
U4-2.5f: Brieftauben 229
Arbeitsblatt U4-1: Wenn-Dann-Labyrinth 231
Arbeitsblatt U4-2: Pseudocode Schritt-für-Schritt 232
Arbeitsblatt U4-3: Line Tracker Testzone 233
Arbeitsblatt U4-4: Nicht reflektierender Rand 234
Arbeitsblatt U4-5: Wenn Linie, nach rechts gehen 235
Arbeitsblatt U4-6: Programmierbare Fernbedienungscodes 236
Einheit 5: Vielseitige Variablen 237
U5-1.1: „Expressions“ 238
U5-1.2: Edisons Lichtsensoren 242
U5-1.2a: Edison die Motte 246
U5-1.2b: Edison, die Kakerlake 248
U5-1.3: Variablen 249
U5-1.3a: Spiralförmige Spinnenfalle 255
U5-1.3b: Fahre ein zufälliges Quadrat 256
U5-1.4 Die Verwendung von Variablen mit Sensordaten 258
U5-1.4a: Edison der Sprinter 260
U5-1.4b: Edison-kontrollierte Flaggenmaschine 261
U5-1.4c: Hey Edison, wo soll ich hin? 265
U5-1.4d: Der Edison-Refrain 272

Arbeitsblatt U5-1: Vier Zeilen 278
Einheit 6: Zeit des Erfinders! 279
U6-1.1: Design-Build-Test-Zyklus erkunden 280
U6-1.1a: Erfinde ein Fantasiewesen 288
U6-1.1b: Erfinde einen Wattebausch-Werfer 289
U6-1.1c: Erfinde einen Einbrecheralarm 290
U6-1.1d: Erfinde eine Mausefalle 291
U6-1.1e: Erfinde einen Kombinationssafe 292
U6-1.2: Lass uns ein Spukhaus erforschen 293
Arbeitsblatt U6-1: Sechs Ideen 303

Einheit 1: Erste Schritte

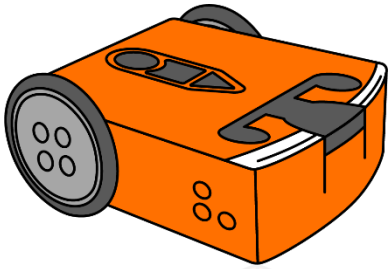


Edison

Name_____

U1-1.1: Dein Edison-Roboter

Das ist Edison, der programmierbare Roboter.



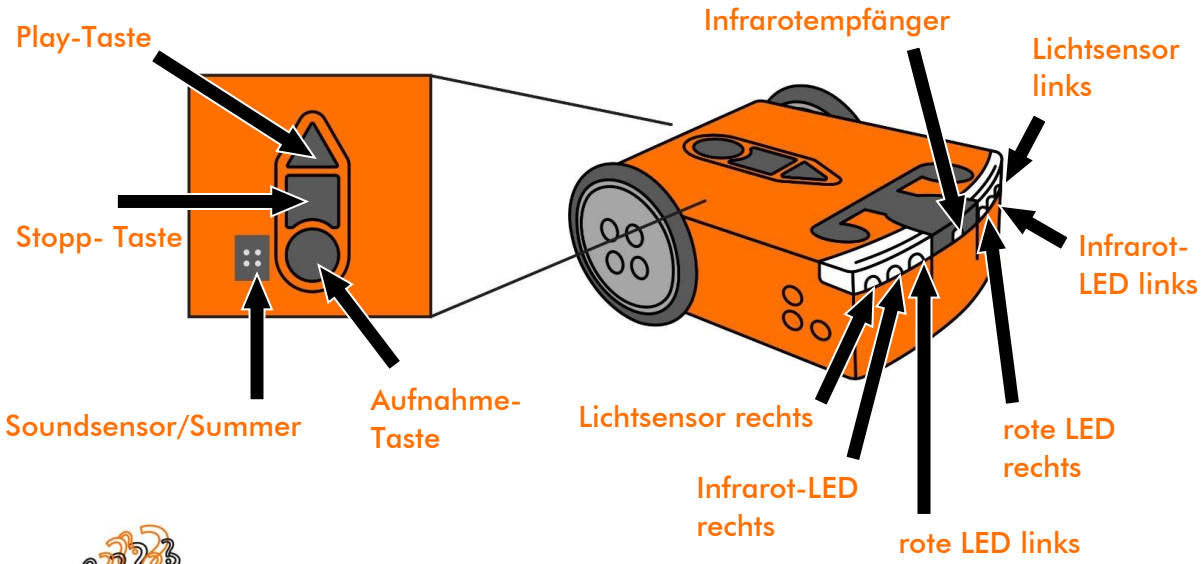
Es gibt eine Menge, was wir mit unseren Edison-Robotern tun können. Wir können den Roboter so programmieren, dass er mit seinen Motoren fährt, mit seinen LED-Leuchten blinkt oder Geräusche macht. Wir können Edison auch benutzen, um Roboterkreationen zu bauen, Labyrinthe zu vervollständigen und vieles mehr!

Bevor wir mit Edison anfangen, müssen wir ein bisschen mehr über den Roboter wissen.

Edison benutzt Sensoren und Motoren, um mit der Welt zu interagieren. Edison hat außerdem drei Knöpfe, einen Netzschalter und mehrere abnehmbare Teile. Zu wissen, wo Edisons Teile sind und was sie bewirken, wird dir helfen, Edison zu benutzen.

Aufgabe 1: Betrachte Edison von oben

Werfe einen Blick auf die Spitze deines Edison-Roboters. Versuche, alle auf dem Bild beschrifteten Teile an deinem Edison-Roboter zu finden.



Woran liegt das?

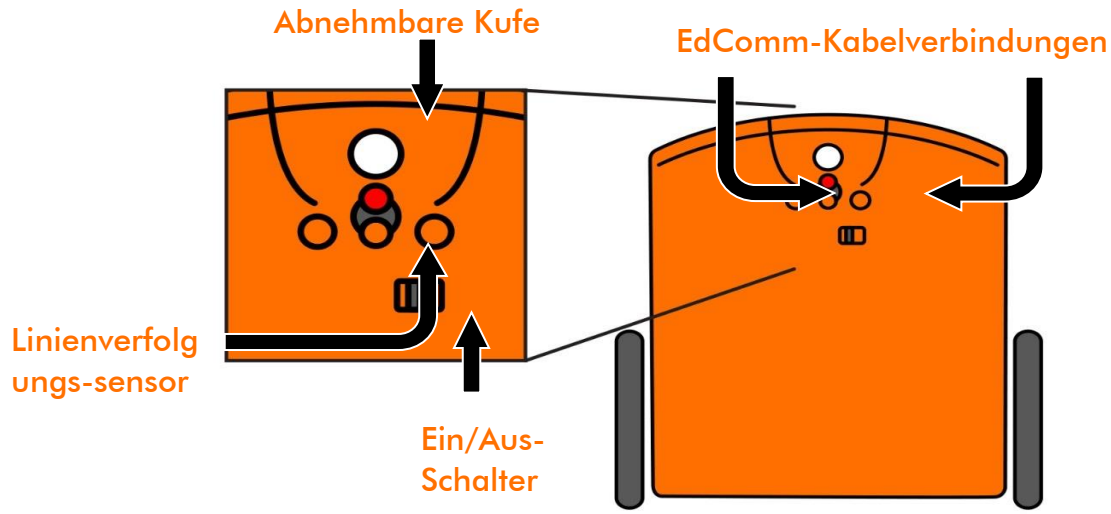
Die Spitze von Edison ist aus durchsichtigem Plastik. Auf diese Weise kann man die elektronischen Komponenten sehen, die Edison funktionieren lassen. Einer der wichtigsten Teile ist das schwarz gefärbte Quadrat, das direkt über der Spitze des 'Play'-Knopfes (Dreieck) sitzt. Kannst du es sehen?

Das ist der Mikrochip des Roboters. Der Mikrochip ist im Grunde ein winziger Computer, der manchmal auch als Mikrocomputer bezeichnet wird. Er enthält die zentrale Verarbeitungseinheit (CPU). Das ist quasi das Gehirn von Edison!

Name_____

Aufgabe 2: Schau dir die Unterseite des Edison-Roboters an

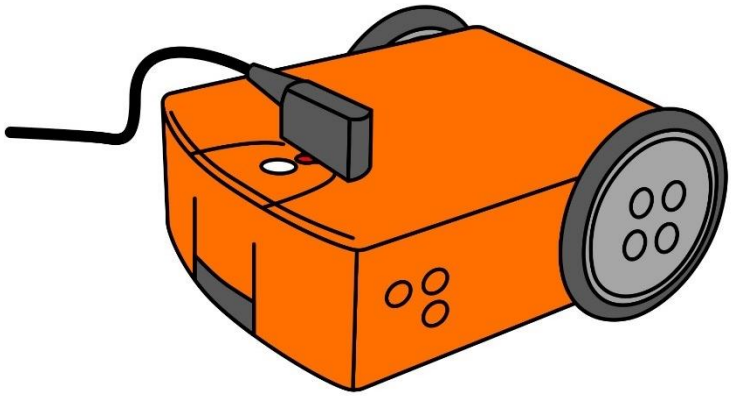
Dreht Edison um. Schau dir das Bild an und versuche, alle auf dem Bild beschrifteten Teile an deinem Edison-Roboter zu finden.



Aufgabe 3: Entferne und befestige die Räder, die Kufe und das EdComm-Kabel

Manchmal möchtest du Edison vielleicht auf verschiedene Weise benutzen, zum Beispiel, indem du den Roboter auf die Seite setzt. Deshalb kann man einige Teile von Edison vom Roboter abnehmen. Beide Räder von Edison können abgenommen werden. Versuche eines der Räder zu entfernen, indem du es geradewegs vom Roboter wegziehst. Schau dir die Buchse an, an der das Rad befestigt ist. Stelle sicher, dass du das Rad wieder einsteckst!

Als Nächstes schau dir Edisons Plastikgleitkufe an. Die Kufe ist das durchsichtige Stück Plastik auf der Unterseite von Edison in der Nähe des Leitungsverfolgungssensors. Die meiste Zeit wirst du die Kufe im Roboter behalten wollen. Die Kufe ist sehr klein und durch das durchsichtige Plastik kann sie schwer zu sehen sein, also sei vorsichtig, wenn du sie entfernst! Du willst die Kufe nicht fallen lassen und verlieren!



Es gibt noch eine weitere Komponente, die wir bei dem Edison-Roboter viel verwenden werden, das EdComm-Kabel.

Du wirst das EdComm-Kabel benutzen, um deine Programme von deinem Programmiergerät, wie z.B. deinem Computer, auf Edison herunterzuladen. Das EdComm-Kabel hat an einem Ende einen Anschluss für Edison, und das andere Ende wird mit der Kopfhörerbuchse deines Computers verbunden.

Zum Üben kannst du versuchen, das EdComm-Kabel an Edison anzuschließen.

Name_____

Aufgabe 4: Edison einschalten

Wann immer wir Edison benutzen wollen, müssen wir den Roboter einschalten. Versuche, Edison jetzt einzuschalten.

1. Was passiert, wenn du den Roboter einschaltest? Beschreibe, was passiert, einschließlich dessen, was du gesehen und gehört hast. Schreibe deine Antwort hier:

_____

_____

_____

_____

_____

Nicht vergessen

Wann immer du Edison fertig benutzt hast, schalte den Roboter wieder aus!

Name_____

U1-1.1a: Bausteine, Blöcke und Edison

Schau dir Edison gut an. Siehst du all die Beulen und Löcher auf der Oberseite, den Seiten und der Unterseite des Roboters?

Wahrscheinlich hast du auch schon solche Stollen gesehen, wie die auf der Oberseite von Edison und auf den Rädern von Edison. Warum glaubst du, dass der Roboter diese Stollen und die Löcher an den Seiten und an der Unterseite von Edison hat?

Das sind alles Verbindungspunkte, durch die sich Edison mit jedem LEGO-Stein-kompatiblen Bausystem verbinden lässt.

Es gibt viele Dinge, die wir mit Edison und verschiedenen Bausystemen bauen können. Bei dieser Aktivität ist es dein Ziel, etwas mit LEGO Steinen und Edison zu bauen.

Was du tun sollst

Hol deinen Edison-Roboter, schnapp dir ein paar Blöcke und lass deiner Kreativität und Fantasie freien Lauf!

Versuche, Edisons Oberseite, Unterseite, Seiten oder Räder mit Blöcken zu versehen. Dekoriere Edison so, wie du es gerne hättest!

Wenn du fertig bist, schreibe eine Beschreibung oder zeichne ein Bild von deinem Edison mit den zusätzlichen Bausteinen. Wie hast du mit Edison gebaut?

Name_____

U1-1.2: Lass uns die Strichcode-Programmierung erforschen

Genau wie alle Roboter und Computer braucht Edison Programme, um zu funktionieren.

Was ist ein Computerprogramm?

Fachjargon

Ein Computerprogramm ist eine Sammlung von Anweisungen, die einen Computer anweisen, eine bestimmte Aufgabe auszuführen.

Jargon ist ein Begriff für die speziellen Wörter oder Ausdrücke, die von Menschen in einer bestimmten Gruppe verwendet werden, wie zum Beispiel eine Art von Arbeit. Jargon ist für Menschen außerhalb dieses Berufs oder dieser Gruppe oft schwer zu verstehen. Computerprogrammierung verwendet einige Wörter und Ausdrücke, die dir jetzt wie 'Jargon' vorkommen könnten - aber diese neuen Vokabeln werden dir bald sehr vertraut sein! Die Jargon-Busterboxen in diesen Lektionen werden dich mit neuen Begriffen bekannt machen.

Okay, lass uns zurück zur Programmierung von Edison kommen.

Edison mit Strichcodes programmieren

Edison kommt mit einigen Programmen, die bereits im Roboter geladen sind. Wir können den Roboter dazu bringen, auf diese Programme zuzugreifen und sie auszuführen, indem wir spezielle Strichcodes verwenden.

Woran liegt das?

Der Mikrochip von Edison hat die Fähigkeit, einige Dinge, wie Programme, zu speichern. Diese Programme werden im Speicher des Roboters abgelegt. Wir können Edison sagen, welche dieser Programme wir ausführen wollen, indem wir über spezielle Strichcodes fahren.

Name_____

Immer wenn du einen der speziellen Strichcodes von Edison benutzt, musst du die gleichen vier Schritte befolgen:

1. Platziere Edison gegenüber dem Strichcode auf der rechten Seite des Strichcodes.
2. Drücke dreimal die Record (runde) Taste.
3. Warte, während Edison vorwärts fährt und den Strichcode einscannt.
4. Drücke einmal die Play-Taste (Dreieck), um das Programm zu starten.

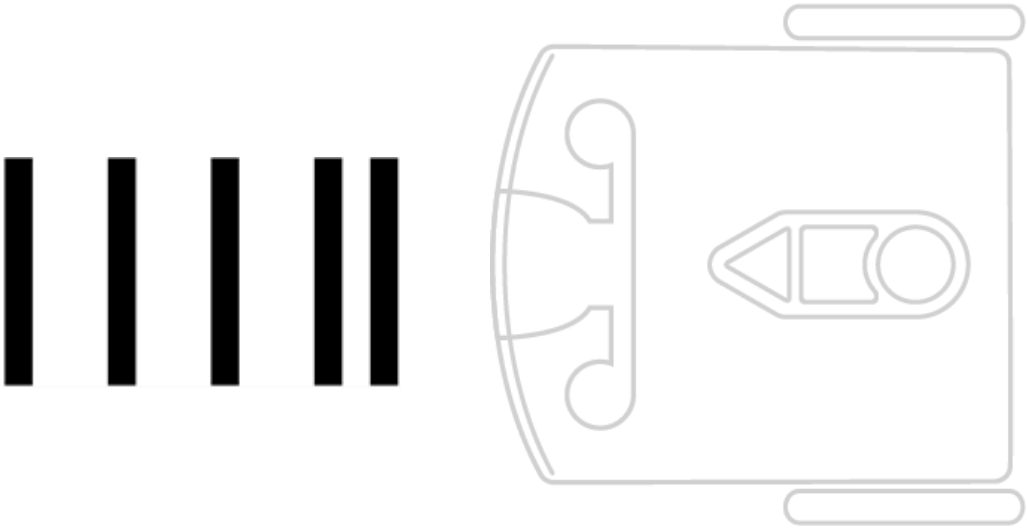
Lass uns versuchen, einige von Edisons Strichcodes zu benutzen.

Name_____

Aufgabe 1: Klatschgesteuertes Fahren

Dieses Programm benutzt Edisons Schallsensor. Der Geräuschsensor kann laute Geräusche erkennen, z.B. wenn man in die Hände klatscht. Dieses Programm sagt Edison, dass er auf ein Klatschen 'hören' soll.

Edison soll den Strichcode lesen.



Nicht vergessen

Um Edison mit einem Strichcode zu programmieren, folge immer diesen Schritten:

1. Platziere Edison gegenüber dem Strichcode auf der rechten Seite des Strichcodes.
2. Drücke dreimal auf die Aufnahmetaste (rund).
3. Warte, während Edison vorwärts fährt und den Strichcode einscannt.
4. Drücke die Play (Dreieck)-Taste einmal, um das Programm zu starten.

Scanne den Strichcode ein und lege Edison dann auf den Boden oder Tisch, bevor du die Play-Taste (Dreieck) drückst. Nachdem du die Play-Taste gedrückt hast, klatsche einmal in die Hände. Edison wird sich nach rechts drehen.

Als Nächstes klatschst du zwei Mal in die Hände. Edison wird vorwärts fahren.

Wenn Edison dein Klatschen nicht erkennen kann, versuche stattdessen mit dem Finger auf die Oberseite des Roboters in der Nähe des Geräuschsensors zu tippen.

Name_____

Aufgabe 2: Hindernissen ausweichen

Dieses Programm verwendet Edisons Infrarotlichtsensor, um Hindernisse auf dem Weg des Roboters zu erkennen und ihnen auszuweichen.

Edison soll den Strichcode lesen.

Bevor du die Play-Taste (Dreieck) drückst, musst du Edison auf den Boden oder Tisch mit einigen Hindernissen legen. Stelle einige Hindernisse für Edison her, indem du Gegenstände um Edison herumlegst. Wähle Objekte, die mindestens so hoch wie Edison und nicht durchsichtig sind. Du kannst auch deine Hände benutzen, um kleine 'Mauern' für Edison zu machen.

Drücke die Play-Taste (Dreieck). Beobachte, was passiert, wenn Edison ein Hindernis entdeckt.

Name_____

Aufgabe 3: Einer Lichtquelle folgen

Dieses Programm benutzt Edisons Lichtsensor, um ein helles Licht zu erkennen und zu verfolgen. Damit dieses Programm funktioniert, brauchst du eine Lichtquelle, eine Taschenlampe oder eine andere Möglichkeit, ein helles Licht zu erzeugen.

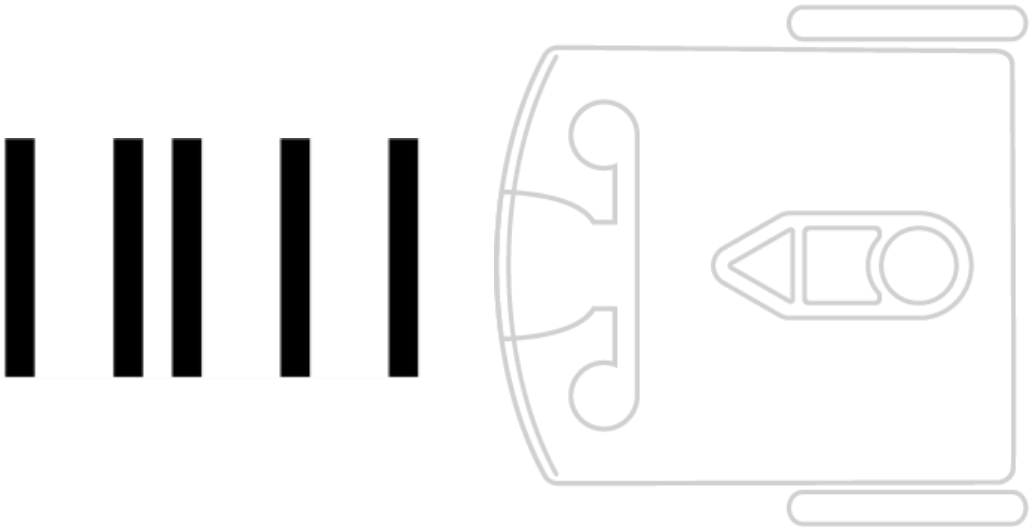
Lass Edison den Strichcode lesen.

Lege Edison auf den Boden oder Tisch und bereite deine Taschenlampe vor, bevor du die Play-Taste (Dreieck) drückst. Richte deine Lichtquelle auf Edison. Der Roboter wird dem hellen Licht folgen.

Name_____

Aufgabe 4: Einer Linie folgen

Dieses Programm benutzt Edisons Linienverfolgungssensor, um eine dunkle Linie zu erkennen und ihr zu folgen. Du brauchst eine dunkle Linie, damit Edison ihr folgen kann. Benutze das Arbeitsblatt U1-1, eine EdMat oder mache deine eigene Linie, der Edison folgen kann.



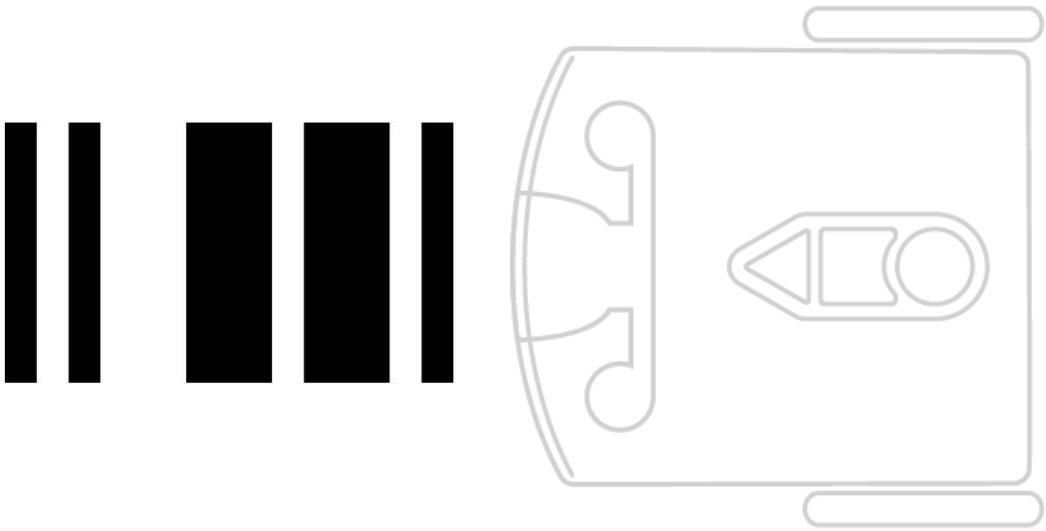
Lass Edison den Strichcode lesen.

Bereite dein Arbeitsblatt vor. Du musst den Roboter auf der weißen Fläche in der Nähe der schwarzen Linie starten. Lege Edison neben die schwarze Linie, aber nicht auf die Linie. Drücke die Play-Taste (Dreieck). Edison wird die Linie finden und ihr folgen.

Name_____

Aufgabe 5: Grenzen abfahren

Dieses Programm benutzt Edisons Linienverfolgungssensor, um dunkle Oberflächen zu erkennen und ihnen auszuweichen. Du brauchst eine Form mit einem dunklen Umriss, um Edison 'einzufangen'. Benutze dein Arbeitsblatt U1-1, dein Arbeitsblatt U1-2, eine EdMat oder mache deine eigene Form, um Edison in die Falle zu locken.



Lass Edison den Strichcode lesen.

Bereite dein Arbeitsblatt vor. Du musst den Roboter auf der weißen Fläche innerhalb der schwarzen Linie starten. Du kannst Edison neben die schwarze Linie setzen, aber nicht auf die Linie. Drücke die Play-Taste (Dreieck). Edison wird innerhalb der dunklen Ränder 'herumhüpfen'.

Name_____

U1-1.2a: Sumo-Ringen

Eines der voreingestellten Programme von Edison ist eigentlich eine Kombination aus zwei anderen Programmen von Edison - Grenzen abfahren und Hinderniserkennung.

Was macht dieses kombinierte Programm? Es ermöglicht es zwei oder mehr Edison-Robotern, Sumo zu ringen!

Der Hinderniserkennungsteil des Programms hilft jedem der Roboter, die anderen Roboter zu finden. Der Linienerkennungsteil des Programms hilft Edison eine Linie zu finden, um den anderen Roboter aus dem Ring zu schlagen.

Du brauchst eine Form mit einem dunklen Umriss, um der Sumo-Ring für die kämpfenden Edison-Roboter zu sein. Benutze dein Arbeitsblatt U1-2, eine EdMat oder baue deinen eigenen Sumoring.

Für diese Aktivität müsst ihr zusammen arbeiten. Scanne den Strichcode mit mindestens zwei Edison-Robotern ein.

Nicht vergessen

Um Edison mit einem Strichcode zu programmieren, folge immer diesen Schritten:

1. Platziere Edison gegenüber dem Strichcode auf der rechten Seite des Strichcodes.
2. Drücke dreimal auf die Aufnahmetaste (rund).
3. Warte, während Edison vorwärts fährt und den Strichcode einscannt.
4. Drücke die Play (Dreieck)-Taste einmal, um das Programm zu starten.

Name_____

Bereite deinen Sumoring vor. Wenn du willst, kannst du die verschiedenen Edison-Roboter mit Etiketten oder durch Anbringen von farbigen Steinen markieren. Setze alle Edison-Roboter in den Ring.

Drücke die Play-Taste (Dreieckstaste) auf allen Robotern gleichzeitig.

Jeder Edison-Roboter fängt an, langsam im Inneren des Rings herumzufahren und nach den anderen Robotern zu suchen. Wenn ein Edison einen anderen Roboter entdeckt, wird er schneller fahren, um ihn zu treffen und zu versuchen, ihn aus dem Ring zu schieben.

Der Edison, der im Ring bleibt, gewinnt!

Name_____

U1-1.2b: Erstelle deinen eigenen Strichcode

Strichcodes mit Edison zu benutzen macht viel Spaß! Die Leute wollen oft wissen, ob sie ihren eigenen Strichcode für Edison machen können.

Denk mal nach. Was machen die Strichcodes? Weißt du noch, was passiert, wenn wir einen speziellen Edison-Strichcode verwenden? Glaubst du, dass es möglich ist, einen eigenen Strichcode zu machen, den Edison lesen kann?

Woran liegt das?

Die Strichcodes sagen Edison einfach, dass er das richtige voreingestellte Programm ausführen soll, wenn die Dreieckstaste gedrückt wird. Die eigentlichen Programme werden in einem Abschnitt des Speichers von Edison gespeichert, der nie geändert wird, so dass es nicht möglich ist, zusätzliche Programme hinzuzufügen, die mit Hilfe der Strichcodes gelesen werden können.

Also, nein, wir können keinen eigenen Strichcode für Edison machen. Aber lass uns so tun, als ob wir es könnten!

Was zu tun ist

Tu so, als könntest du dir einen eigenen Strichcode für Edison machen, der ein Programm ausführen würde. Was würde dieses Programm Edison sagen, was es tun soll?

1. Schreibe eine Beschreibung deines vorgetäuschten Programms. Erkläre, was Edison tun würde, wenn der Roboter dein Programm ausführen könnte.

_____

_____

_____

_____

2. Zeichne deinen erfundenen Strichcode.

Name_____

U1-1.2c: TV-Fernbedienung Strichcodes

Es gibt noch einen weiteren Satz spezieller Strichcodes, die wir mit Edison verwenden können. Diese Strichcodes ermöglichen es Edison, auf Knopfdruckbefehle von deiner TV- oder DVD-Fernbedienung zu reagieren.

Woran liegt das?

Die ferngesteuerten Strichcodes sind eine besondere Art von Strichcode. Mit diesen Strichcodes kann Edison einen Code von einer Fernbedienung speichern. Edison kann diesen Code dann später referenzieren.

Im Gegensatz zu anderen Edison-Strichcodes aktivieren diese Strichcodes nicht von sich aus ein Programm in Edison. Stattdessen weisen diese Strichcodes Edison an, nach einem Fernbedienungscode zu suchen. Wenn Edison einen von ihm erkannten Fernbedienungscode erkennt, führt der Roboter eine vorprogrammierte Aktion aus.

Wenn du diese Strichcodes zusammen mit einer TV- oder DVD-Fernbedienung verwendest, kannst du Edison wie ein ferngesteuertes Auto herumfahren!

Aufgabe 1: Plane deine Fernbedienungsverbindung

Schau dir die acht Strichcodes auf dem Arbeitsblatt U1-3 an. Sechs der Strichcodes weisen Edison an, Programme laufen zu lassen, die steuern, wie Edison sich bewegen wird. Die letzten beiden Strichcodes sagen Edison, dass er Programme laufen lassen soll, die Geräusche machen. Du musst jedes Strichcode-Programm mit einer anderen Taste auf deiner Fernbedienung verbinden.

Um es einfacher zu machen, Edison mit der Fernbedienung zu steuern, musst du die Aktion des Programms mit einer Taste auf der Fernbedienung abstimmen, die Sinn macht. Zum Beispiel könntest du einen Pfeil nach oben" (wie auf Lautstärke erhöhen") für das Vorwärtsfahrtprogramm verwenden.

Schau dir die Fernbedienung an, die du benutzt, und entscheide dich für eine Taste für jedes Programm. Schreibe deine Tastenwahl für jedes Programm auf:

Name_____

Programm	Knopf auf der Fernsteuerung
Vorwärts fahren	
Rückwärts fahren	
Links drehen	
Rechts drehen	
Links abbiegen	
Rechts abbiegen	
Spiele Ton	
Spiele Melodie	

Name_____

Aufgabe 2: Programmiere Edison mit der Fernbedienung

Programmiere Edison mit jeder TV-Fernbedienung, einen Strichcode nach dem anderen.

Nicht vergessen

Um Edison mit einem Strichcode zu programmieren, folge diesen Schritten:

1. Platziere Edison gegenüber dem Strichcode auf der rechten Seite des Strichcodes.
2. Drücke dreimal auf die Aufnahmetaste (rund).
3. Warte, während Edison vorwärts fährt und den Strichcode einscannt.
4. Wenn du einen Edison mit einer TV-Fernbedienung koppelst, musst du einen anderen letzten Schritt machen:
5. Drücke die Taste auf deiner TV-Fernbedienung, die du mit der Aktion des Strichcodes abgleichen möchtest.

Wenn du diese Fernbedienungs-Strichcodes benutzt, drückst du nicht die Play-Taste (Dreieck) auf Edison. Stattdessen drückst du die Taste, die du gerade gepaart hast, auf der Fernbedienung. Wenn Edison das Signal des Fernbedienungscodes erkennt, wird der Roboter die Aktion dieses Strichcodes ausführen.

Probiert es aus!

Versuche Edison mit deiner Fernbedienung zu steuern, indem du die Aktionen verwendest, die du mit den Strichcodes der Fernbedienung programmiert hast.

Name_____

U1-1.2d: Edison-Fußball

Du kannst eine TV- oder DVD-Fernbedienung zusammen mit den Strichcodes der TV-Fernbedienung benutzen, um die Bewegung deines Edison-Roboters zu steuern. Kannst du Edison gut genug steuern, um Fußball gegen einen anderen Edison-Roboter zu spielen? Schnapp dir einen Gegner, baue das Feld auf und spiele ein Spiel, um den Edison-Fußballmeister zu ermitteln!

Was zu tun ist

Für diese Aktivität müsst ihr zusammen arbeiten. Stellt sicher, dass jeder Spieler einen Edison-Roboter und eine Fernbedienung hat. Alle Edison-Roboter, die im Spiel spielen, müssen mit den Strichcodes der TV-Fernbedienung programmiert werden. Benutze die Strichcodes auf dem Arbeitsblatt U1-3.

Tipp!

Du solltest andere Befehle als die der anderen Spieler verwenden. Sonst kann es passieren, dass du am Ende deine Gegner kontrollierst!

Außerdem musst du ein Spielfeld für die Roboter, Tore und einen Ball einrichten.

Wie kannst du das Spielfeld gestalten? Welche Ballgröße wird am besten funktionieren?

Experimentiere, um zu sehen, was funktioniert!

Name_____

U1-1.2e: Baue und kontrolliere den EdTank

Mit den Strichcodes der TV-Fernbedienung kannst du deinen Edison-Roboter so steuern, dass er sich auf verschiedene Arten bewegt. Die ferngesteuerten Strichcodes, die Edisons Bewegung steuern, steuern eigentlich die Motoren des Roboters. Was passiert, wenn an den Motoren keine Räder angebracht sind, sondern etwas anderes?

Was zu tun ist

In dieser Aktivität wirst du den EdTank bauen und kontrollieren.

Der EdTank ist ein ferngesteuerter Panzer, den du herumfahren kannst. Du kannst auch einen zweiten Edison-Roboter benutzen, um mit einer Kanone, mit der du ein Gummiband abfeuern kannst, eine obere Schicht des Panzers zu bauen.

Benutze diesen Link

Gehe zu meetedison.com/Inhalt/EdCreate/EdBuild-EdTank-instructions.pdf

Dieser Link führt euch zu den Schritt-für-Schritt-Anleitungen für den Bau und die Programmierung des EdTanks.

Probiere es aus!

Wenn du den EdTank gebaut und programmiert hast, versuche ihn herumzufahren!

1. Bemerkst du Unterschiede in der Fahrweise des EdTank im Vergleich zu Edison, der normalerweise fährt, wenn der Roboter nur seine Räder montiert hat? Denk darüber nach, was die Unterschiede verursachen könnte, die dir auffallen. Was könnte die Fahrweise des EdTank beeinflussen?

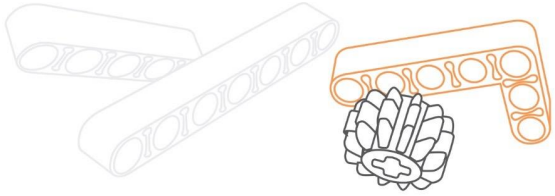
_____

_____

_____

_____

_____



Name_____

U1-1.2f: Challenge: Baue und kontrolliere den EdDigger

Mit den Strichcodes der TV-Fernbedienung kannst du deinen Edison-Roboter so steuern, dass er sich auf verschiedene Arten bewegt. Die ferngesteuerten Strichcodes, die Edisons Bewegung steuern, steuern eigentlich die Motoren des Roboters. Was passiert, wenn an den Motoren keine Räder angebracht sind, sondern etwas anderes?

Was zu tun ist

In dieser Aktivität wirst du den EdDigger bauen und kontrollieren.

Der EdDigger ist ein ferngesteuerter Bagger, mit einer Schaufel, die du herumfahren kannst. Die Baggerschaufel des EdDigger kann heben und senken und kleine Gegenstände tragen.

Benutze diesen Link

Gehe zu meetedison.com/Inhalt/Erstelle/EdBuild-EdDigger-Anweisungen.pdf

Dieser Link führt dich zu den Schritt-für-Schritt-Anleitungen für den Bau und die Programmierung des EdDigger.

Probier es aus!

Sobald du den EdDigger gebaut und programmiert hast, kannst du versuchen, ihn herumzufahren! Achte darauf, dass du auch einige Gegenstände aufschaufelst!

Kannst du die Baggerschaufel reibungslos bedienen? Was müsst ihr tun, damit der EdDigger Objekte schaufelt? Was müsst ihr tun, damit der EdDigger Gegenstände fallen lässt, die er mit sich führt?

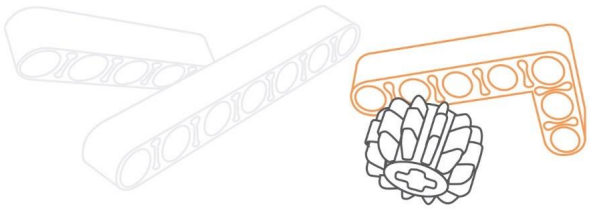
_____

_____

_____

_____

_____



Name_____

U1-1.2g: Baue und kontrolliere die EdRoboClaw

Mit den Strichcodes der TV-Fernbedienung kannst du deinen Edison-Roboter so steuern, dass er sich auf verschiedene Arten bewegt. Die ferngesteuerten Strichcodes, die die Bewegung des Edison-Roboters steuern, steuern eigentlich die Motoren des Roboters. Was passiert, wenn an den Motoren keine Räder angebracht sind, sondern etwas anderes?

Was zu tun ist

In dieser Aktivität wirst du die EdRoboClaw bauen und kontrollieren.

Der EdRoboClaw ist ein ferngesteuerter Roboterarm mit einer beweglichen Basis, um die du herumfahren kannst. Der Roboterarm des EdRoboClaw kann sich öffnen und schließen, um Gegenstände aufzunehmen und zu tragen.

Benutze diesen Link

Gehe zu meetedison.com/Inhalt/EdCreate/EdBuild-EdRoboClaw-Anweisungen.pdf

Über diesen Link kommst du zu den Schritt-für-Schritt-Anleitungen für den Bau und die Programmierung des EdRoboClaw.

Probier es aus!

Wenn du den EdRoboClaw gebaut und programmiert hast, versuche ihn herumzufahren! Vergewissere dich, dass du auch einige Gegenstände aufheben und tragen kannst!

1. Experimentiere mit verschiedenen Gegenständen mit der EdRoboClaw. Welche Arten von Gegenständen kannst du tragen? Welche Arten von Gegenständen haben nicht funktioniert? Überlegt euch, was die Gegenstände, die gut funktioniert haben, miteinander gemeinsam haben. Was macht einen guten Gegenstand für die EdRoboClaw aus?

_____

_____

_____

_____

_____

Name_____

U1-2.1: Lass uns die EdScratch-Umgebung erkunden

Eines der besten Dinge an Edison ist, dass du deine eigenen Programme für deinen Roboter erstellen kannst! Um ein Programm für Edison zu schreiben, müssen wir eine spezielle Software benutzen.

Fachjargon

Alle Computer haben zwei Hauptbestandteile: Hardware und Software.

Hardware ist der physische Teil eines Computers (oder Roboters).

Software ist die Gesamtheit der Programme und Anwendungen, die die Hardware, wie einen Computer oder einen Roboter, zum Laufen bringt.

Die Software, die wir mit Edison benutzen werden, ist eine Roboter-Programmiersprache.

Fachjargon

Eine Programmiersprache ist ein Satz von Regeln und Anweisungen, mit denen Computerprogramme geschrieben werden. EdScratch ist eine Programmiersprache, die speziell für die Programmierung von Edison-Robotern entwickelt wurde.

Die Programmiersprache, die wir benutzen werden, heißt EdScratch. Lass uns ein bisschen über die EdScratch-Programmiersprache lernen.

Name_____

Aufgabe 1: Schaut euch EdScratch an

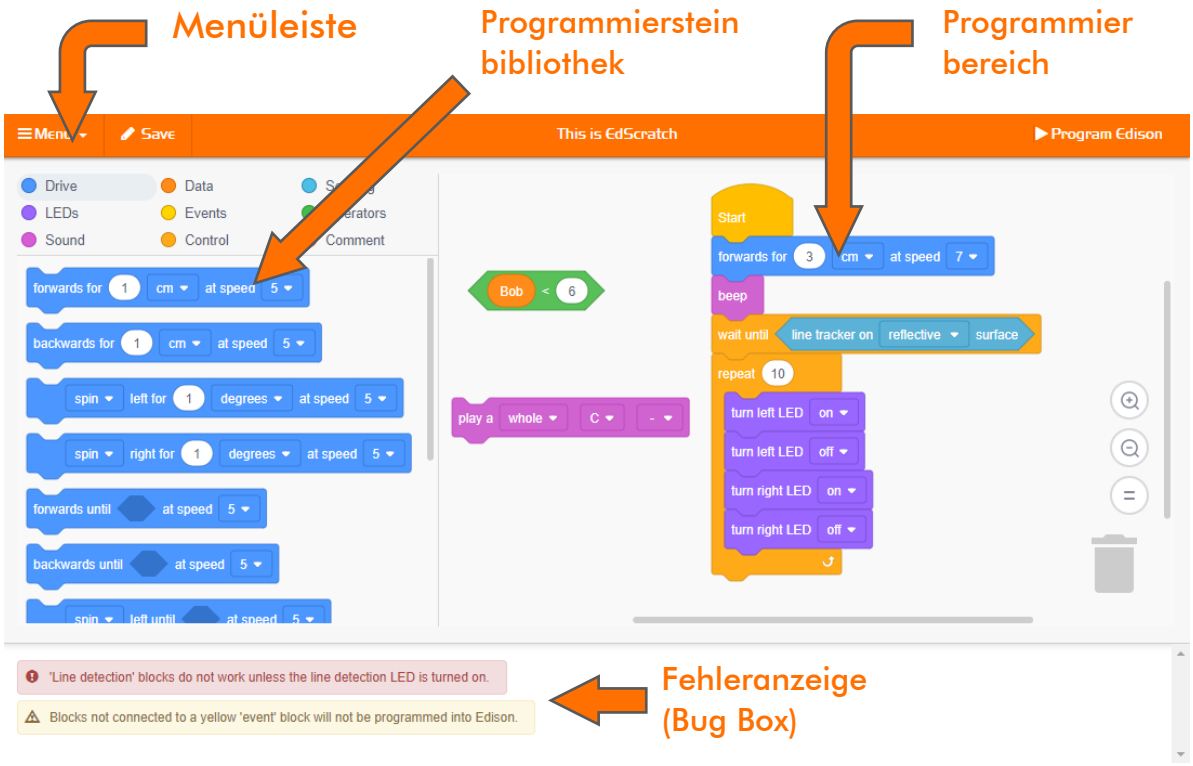
Du kannst online auf EdScratch zugreifen.

Benutze diesen Link

Gehe zu www.edscratchapp.com

Wann immer du Edison mit EdScratch programmieren möchtest, musst du zur EdScratch App gehen.

Programmierbereich Hier siehst du, wie die EdScratch-Umgebung aussieht:



Name_____

Die EdScratch-Programmierumgebung besteht aus vier Hauptteilen:

Programmiersteinbibliothek
Alle Blöcke, die du benutzen kannst, befinden sich in der Programmiersteinbibliothek. Um einen Block zu benutzen, wähle ihn aus der Palette aus und ziehe ihn in den Programmierbereich.

Programmierbereich
Der große Bereich, in dem du Blöcke zu Programmen zusammenfügen kannst, wird Programmierbereich genannt. Ziehe Blöcke von der Programmiersteinbibliothek in diesen Bereich, um sie in deinem Programm zu verwenden.

Die Menüleiste
Optionen wie 'Speichern' und 'Laden' werden über die Menüleiste aufgerufen. Die Menüleiste hat auch die Schaltfläche 'Program Edison'.

Bug-Box (Fehleranzeige)
Unter der Programmiersteinbibliothek und dem Programmierbereich befindet sich die Bug-Box. Warnmeldungen werden in der Bug-Box angezeigt.

Schau dir EdScratch auf deinem Computer an. Finde jeden der vier Hauptteile der EdScratch-Umgebung.

Name_____

Aufgabe 2: Lade und downlade das Testprogramm

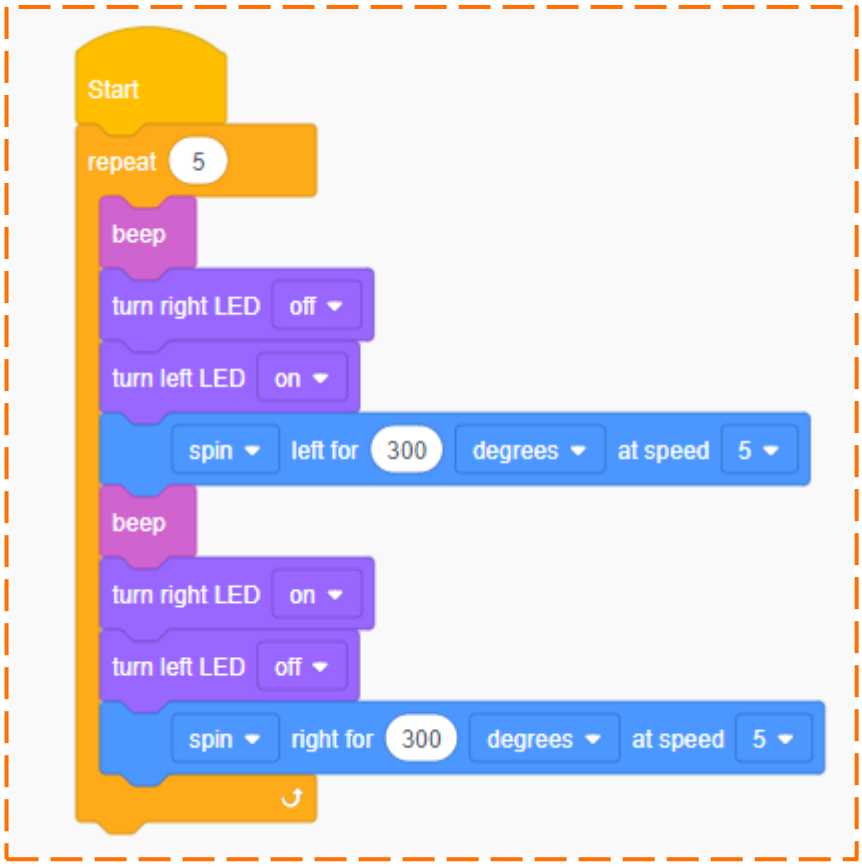
EdScratch hat bereits einige Demoprogramme geschrieben. Versuche das Demoprogramm namens Test_program zu laden und herunterzuladen.

Lade das Test_program Demoprogramm

Um das Test_program zu laden, folge diesen Schritten:

1. Gehe in EdScratch in die Menüleiste und wähle das Drop-Down-Menü. Suche und wähle die Option namens Load Demos. Dies wird ein Pop-up Fenster mit allen Demoprogrammen öffnen.
2. Finde und wähle das Programm namens Test_program. Das Programm wird im Programmierbereich geladen.

So sieht das Test_program aus:



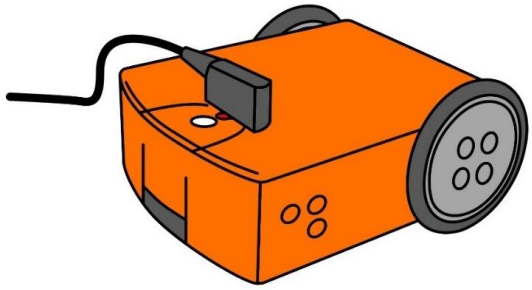
Sobald das Programm im Programmierbereich geladen ist, kannst du es auf deinen Edison-Roboter herunterladen.

Name_____

Test_program auf Edison herunterladen

Wann immer du ein Programm von EdScratch auf Edison herunterladen möchtest, musst du diese Schritte befolgen:

1. Verbinde Edison mit deinem Computer über das EdComm-Kabel.
2. Stelle sicher, dass die Lautstärke auf dem Computer ganz aufgedreht ist.
3. Drücke **einmal** den Aufnahmeknopf (rund) auf Edison.
4. Gehe zur Menüleiste in EdScratch und klicke auf den "Program Edison"-Knopf.
5. Ein Pop-up-Fenster wird sich öffnen. Sobald das Programm fertig ist, erscheint am unteren Rand des Pop-up-Fensters eine Schaltfläche namens Program Edison.
6. Klicke auf den "Program Edison"-Button in dem Pop-up-Fenster.



Woran liegt das?

Edison kann die Blöcke in EdScratch nicht so verstehen, wie sie auf deinem Computerbildschirm aussehen. Die Blöcke müssen in ein Format gebracht werden, das Edison versteht, bevor das Programm heruntergeladen werden kann. Das kann ein bisschen Zeit in Anspruch nehmen.

Deshalb kann es eine Weile dauern, bis der Program-Edison-Button im Pop-up-Fenster erscheint.

Du wirst hören, wie das Programm nach Edison heruntergeladen wird. Sobald der Download abgeschlossen ist, gibt Edison den Piepton 'Erfolg' aus. Trenn Edison nicht vom Netz, bevor du den Piepston hörst!

Name_____

Woran liegt das?

Edison lässt dich wissen, ob das Programm korrekt heruntergeladen wird, indem er den 'Erfolg'-Piepston gibt. Dies ist das gleiche Geräusch, das du hörst, wenn du Edison zum ersten Mal einschaltest.

Es gibt noch ein anderes Geräusch, das Edison möglicherweise macht, wenn ein Programm nicht korrekt heruntergeladen wird. Wir nennen dies den 'Fail'-Ton. Es bedeutet, dass etwas schief gelaufen ist, als das Programm versucht hat, herunterzuladen. Wenn Edison dieses Geräusch macht, versuche den Download erneut zu starten.

Nachdem du den Piepston von Edison hörst, zieh den Stecker des Roboters aus dem EdComm-Kabel. Drücke die Play-Taste (Dreieck) einmal, um das Programm zu starten.

Probiere es aus!

Lade das Test_Program-Demoprogramm in EdScratch. Lade das Programm herunter und starte es mit deinem Edison-Roboter. Beantworte dann die folgenden Fragen.

1. In welchem Teil der EdScratch-Umgebung befindet sich die Schaltfläche Program Edison?

_____

_____

2. Wie viele Warnmeldungen gibt es in der Bug-Box, wenn du das Test_Program lädst?

_____

_____

3. Was macht der Roboter, wenn du das Test_Program ausführst? Beschreibe, was passiert.

_____

_____

_____

_____

_____

Name_____

U1-2.1a: Lade ein weiteres Programm herunter!

Es gibt mehrere Demoprogramme in EdScratch. Wähle ein anderes Programm als Test_program aus der Liste der Demoprogramme.

Versuche das Demoprogramm deiner Wahl herunterzuladen und auszuführen, um zu sehen, was dieses Programm tut.

1. Wie war der Name des Programms, das du ausgewählt hast?

_____

2. Was hast du erwartet, was das Programm tun soll? Hat das Programm das getan, was du erwartet hast?

_____

_____

_____

3. Schau dir das Programm an, das du in EdScratch ausgewählt hast. Denke darüber nach, was der Roboter macht, wenn du das Programm ausführst. Was fällt dir auf? Wie verhalten sich die Blöcke im Programm zu dem, was der Roboter macht, wenn du das Programm in Edison ausführst?

_____

_____

_____

_____

_____

Name_____

U1-2.1b: Ist EdScratch = Scratch?

Kommt dir EdScratch bekannt vor? Möglicherweise, besonders wenn du irgendwelche Projekte mit der Programmiersprache Scratch gemacht hast.

Woran liegt das?

EdScratch sieht Scratch sehr ähnlich. Das ist Absicht! Tatsächlich wurde EdScratch mit Scratch als Basis gebaut.

Also, wenn EdScratch mit Scratch als Basis erstellt wurde, bedeutet das, das EdScratch dasselbe wie Scratch ist?

Nein!

EdScratch und Scratch sind verschiedene Programmiersprachen. Sie haben zwar einige Gemeinsamkeiten, aber es gibt auch eine Menge Unterschiede zwischen den beiden Sprachen. Das Wichtigste ist, dass du deinen Edison-Roboter nicht mit Scratch programmieren kannst. Dafür musst du EdScratch verwenden!

Welche anderen Dinge sind bei EdScratch und Scratch unterschiedlich? Was haben diese beiden Programmiersprachen gemeinsam?

Probier es aus!

Werfe einen Blick auf Scratch.

Vergleiche es mit EdScratch. Was ist das Gleiche und was ist anders?

Benutze diesen Link

Du kannst Scratch über diesen Link besuchen www.scratch.mit.edu

1. Finde drei Dinge über Scratch und EdScratch, die gleich sind. Beschreibe jedes einzelne.

_____

_____

_____

2. Finde drei Dinge, die in Scratch und EdScratch unterschiedlich sind. Beschreibe jedes einzelne.

_____

_____

_____

Name_____

U1-2.2: Warnmeldungen

Einige Programmiersprachen haben besondere Eigenschaften, um die Benutzung dieser Sprache zu erleichtern. Ein Beispiel dafür ist die Bug-Box in EdScratch.

Manchmal, wenn wir ein Programm für Edison in EdScratch schreiben, stimmt etwas nicht ganz. Wenn dies passiert, erscheint eine Warnmeldung in der Bug-Box.

Nicht vergessen

Die Fehlerbox befindet sich unter der Blockpalette und dem Programmierbereich in EdScratch.

Es gibt zwei Arten von Warnmeldungen: gelbe Warnmeldungen und rote Warnmeldungen.

Woran liegt das?

Gelbe Warnmeldungen sind Warnmeldungen. Hier sagt EdScratch "Achtung! Das funktioniert vielleicht nicht so, wie du es dir wünschst". Du kannst ein Programm auch dann herunterladen, wenn in der Bug-Box gelbe Meldungen angezeigt werden.

Rote Warnmeldungen sind wie 'Stopp'-Meldungen. Diese Meldungen sind EdScratch-Meldungen wie "Entschuldigung! Dieses Programm wird für Edison keinen Sinn machen". Wenn es rote Warnmeldungen in der Bug-Box gibt, kannst du das Programm nicht nach Edison herunterladen.

Immer wenn du Programme für Edison schreibst, ist es eine gute Idee, die Bug-Box zu überprüfen, bevor du versuchst, das Programm herunterzuladen. Die Warnmeldungen können dir helfen, dein Programm zu reparieren!

Name_____

Probiere es aus!

In EdScratch findest und lädst du das Demoprogramm namens Warning_messages_demo.

Nicht vergessen

Um zu EdScratch zu gelangen, gehe zu www.edscratchapp.com

Gehe zur Menüleiste und wähle das Drop-Down-Menü aus. Suche und wähle die Option „Load Demos". Daraufhin öffnet sich ein Pop-up-Fenster mit allen Demoprogrammen. Finde und lade das Programm namens Warning_messages_demo.

Sobald das Programm in EdScratch geladen ist, beantworte die folgenden Fragen.

1. Versuche, dieses Programm auf deinen Edison-Roboter herunterzuladen. Was passiert dann? Funktioniert es? Warum oder warum nicht?

_____

_____

_____

2. Lies die rote Nachricht in der Bug-Box. Schau dir das Programm an. Kannst du das Problem beheben? Beschreibe, was du getan hast, um die rote Nachricht zu beheben.

_____

_____

_____

3. Lies die gelbe Nachricht in der Bug-Box. Schau dir das Programm an. Wenn du das Programm herunterlädst, während die gelbe Nachricht dort steht, welche Blöcke werden dann nicht in Edison programmiert?

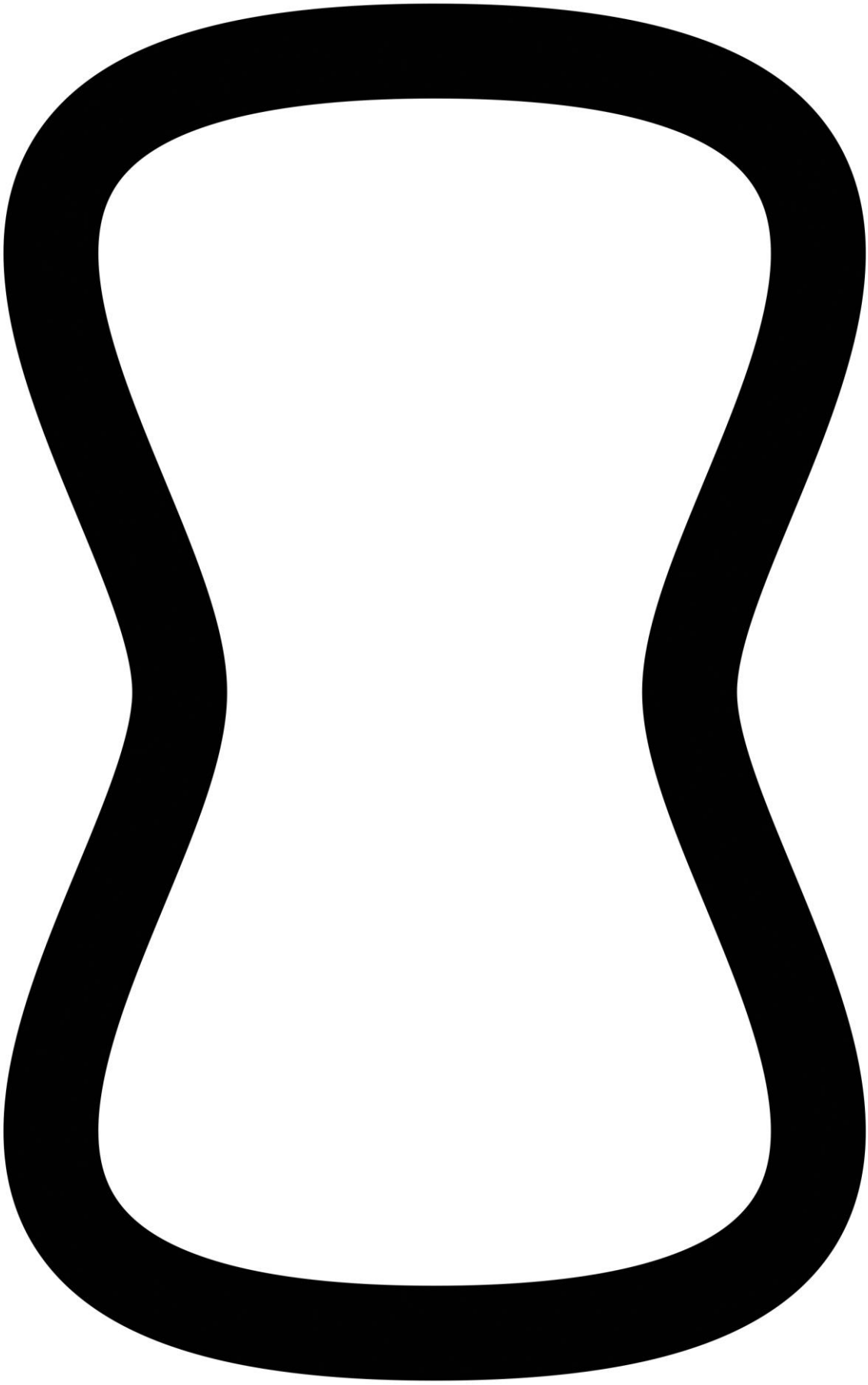
_____

_____

_____

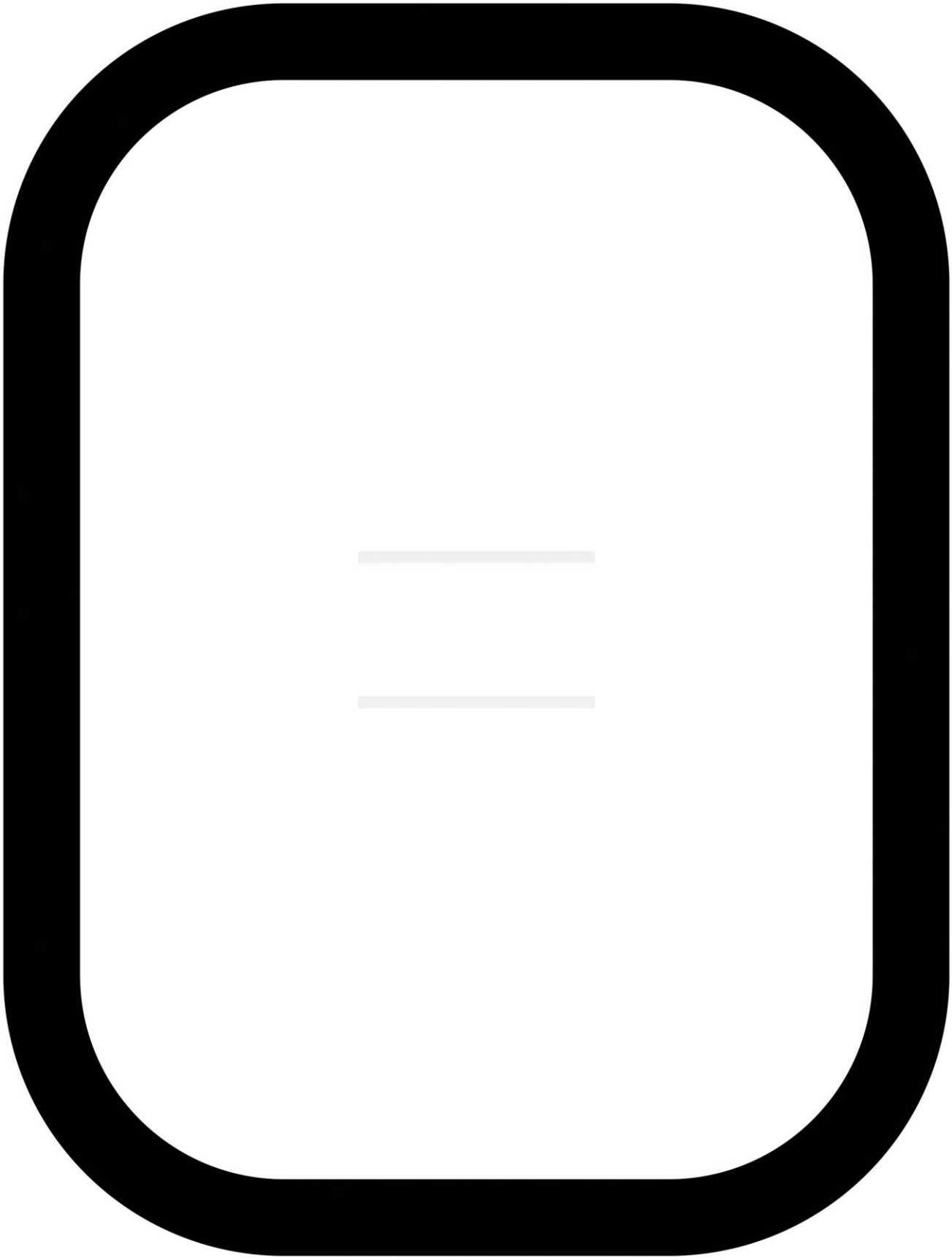
Name_____

Arbeitsblatt U1-1: Grenze der Linienverfolgung



Name_____

Arbeitsblatt U1-2: Sumo-Ring



Name_____

Arbeitsblatt U1-3: TV-Fernbedienungen

Vorwärts fahren

Rückwärts fahren

Nach links drehen

Rechts drehen

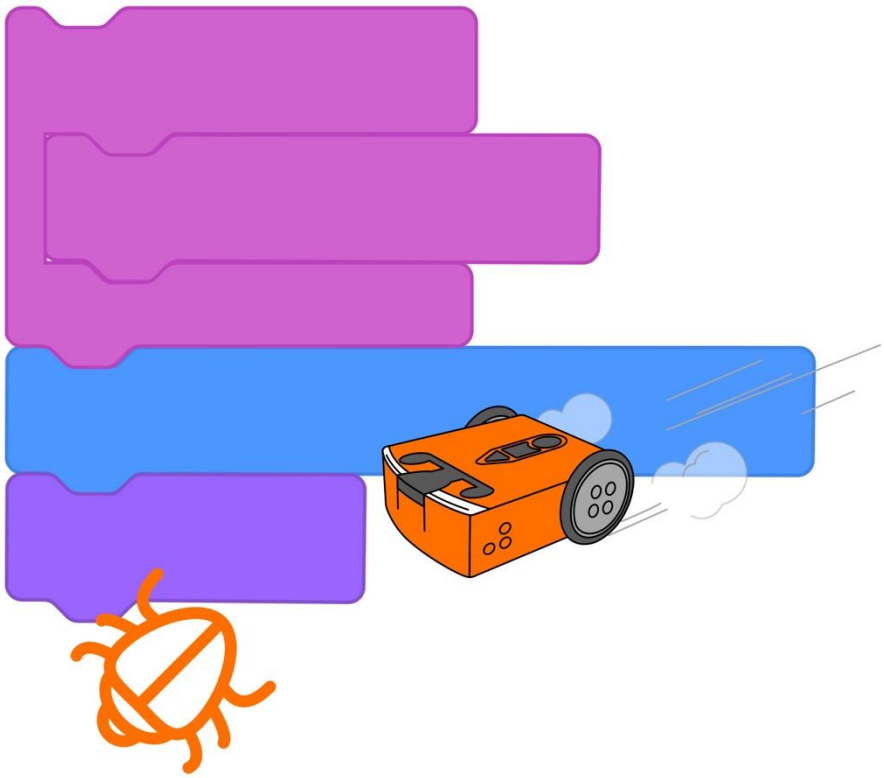
Links abbiegen

Rechts abbiegen

Ton abspielen

Melodie abspielen

Einheit 2: Bewegung!



Edison

Name_____

U2-1.1: Lasst uns erforschen, wie Computer „denken“.

Stell dir einen Computer vor. Jetzt stelle dir eine Person vor. Wen hältst du für klüger?

Das ist ein bisschen eine Fangfrage. Ob du es glaubst oder nicht, die meisten Computer können ohne die Hilfe von Menschen nichts machen.

Woran liegt das?

Sowohl Computer als auch Menschen können Anweisungen befolgen, aber die Menschen können auch selbstständig denken. Wir können lernen und verändern, was wir tun, basierend auf neuem Wissen.

Die meisten Computer können das jedoch nicht tun. Ein Edison-Roboter zum Beispiel kann nicht selbstständig denken. Er kann nur Anweisungen befolgen. Woher kommen diese Anweisungen? Von einem Menschen wie dir kommen diese Anweisungen? Von einem Menschen wie dir!

Menschen geben Computern Anweisungen, indem sie ihnen Computerprogramme geben. Um ein gutes Computerprogramm für unseren Edison-Roboter oder irgendeinen Computer zu machen, müssen wir dieses Programm so schreiben, dass der Computer es verstehen kann. Um das zu tun, müssen wir versuchen, über die Dinge so zu denken, als ob wir ein Computer wären.

Nicht vergessen

Ein Computerprogramm ist eine Sammlung von Anweisungen, die einem Computer mitteilen, eine bestimmte Aufgabe auszuführen.

Diese Art des Denkens nennt man computergestütztes Denken („computational thinking“).

Name_____

Fachjargon

Computergestütztes Denken bedeutet, über ein Problem oder eine Aufgabe nachzudenken, ähnlich wie ein Computer denkt. Es ist ein Weg, Probleme logisch durchzuarbeiten, sie in kleinere Stücke zu zerlegen, Muster zu finden und dann die Informationen zu nutzen, um eine Schritt-für-Schritt-Lösung zu finden.

Mit anderen Worten, computergestütztes Denken ist eine Art und Weise, Informationen zu planen, Probleme zu lösen und zu analysieren, ähnlich wie es ein Computer tut.

Immer wenn du ein Programm für deinen Edison-Roboter schreiben willst, musst du rechnergestütztes Denken anwenden, um herauszufinden, was zu tun ist. Wenn du lernst, auf eine Weise zu denken, die für Edison Sinn macht, kannst du dem Roboter Anweisungen geben, damit er das tut, was du willst.

Eines der wichtigsten Dinge bei der Erteilung von Anweisungen an Edison ist die Reihenfolge, in der du die Anweisungen gibst.

Die Wichtigkeit, Schritt für Schritt vorzugehen

Computer, einschließlich Edison-Roboter, sind sehr gut darin, den Anweisungen zu folgen, die wir ihnen als Computerprogramme geben. Tatsächlich wird ein Edison-Roboter die Anweisungen in einem Programm genauso befolgen, wie sie geschrieben sind. Deshalb ist einer der wichtigsten Teile des rechnerischen Denkens die Verwendung von Sequenzen.

Fachjargon

Sequenz bedeutet, der Reihe nach zu gehen, Schritt für Schritt.

Stell dir vor, du willst einen Kuchen backen. Du könntest ein Rezept in einem Kochbuch nachschlagen. Um den Kuchen zu backen, würdest du dann jeden Schritt nacheinander ausführen. Das ist Sequenz!

Name_____

Immer wenn du ein Programm für Edison schreibst, musst du die Sequenz auf die gleiche Weise verwenden. Du musst Edison genau sagen, was er tun soll, und zwar genau in der Reihenfolge, in der der Roboter jeden Schritt ausführen soll.

Aufgabe 1: Folge Schritt für Schritt

Wenn dein Lehrer dir sagt, du sollst zur Tür gehen, was musst du tun, um dorthin zu gelangen? Du denkst wahrscheinlich nicht darüber nach, wie viele Schritte du machen musst. Du tust es einfach! Wenn dir ein Schreibtisch im Weg steht, drehst du dich einfach um und gehst um ihn herum.

So funktioniert ein Roboter nicht. Um deinen Roboter zur Tür zu bringen, müsstest du ihm sehr sorgfältige Anweisungen geben, in denen jeder Schritt einzeln erklärt wird. Mit anderen Worten, du müsstest dem Roboter jede Aktion, die er ausführen soll, der Reihe nach erklären.

Wenn man darüber nachdenkt, so etwas der Reihe nach zu tun, braucht man etwas Übung. Die Menschen sind so gut darin, Dinge einfach nur zu „tun", dass wir normalerweise nicht darüber nachdenken, was wir tun.

Versuche ein paar genaue Schritt-für-Schritt-Anweisungen zu befolgen, um zu sehen, wie es sich anfühlt. Benutze das Arbeitsblatt U2-1, um die folgenden Fragen zu beantworten.

1. Beginne auf der Eistüte mit dem Zeigen auf das Herz. Wende dich nach rechts. Bewege dich 2 Quadrate vorwärts. Wo bist du?

_____

2. Beginne auf dem Pandabären und zeige auf das Fahrrad. Bewege dich 1 Quadrat rückwärts. Wende dich nach links. Bewege dich 2 Quadrate vorwärts. Wende dich nach rechts. Bewege dich 1 Quadrat vorwärts. Wo bist du?

_____

3. Beginne auf dem Stern und zeige auf die Katze. Wende dich nach links. Wende dich wieder nach links. Bewege dich 2 Quadrate rückwärts. Wende dich nach rechts. Bewege dich 1 Quadrat vorwärts. Nach rechts drehen. Bewege dich 1 Feld vorwärts. Wende dich nach links. Bewege dich 2 Felder rückwärts. Wo bist du?

_____

Name_____

Aufgabe 2: Gib Schritt-für-Schritt-Anweisungen

Lasst uns üben, genaue Anweisungen zu geben, indem wir jeden Gegenstand Schritt für Schritt beschreiben. Benutze das Arbeitsblatt U2-1, um die folgenden Fragen zu beantworten.

Tipp!

Benutze diese Befehle, um deine Antworten zu schreiben:

vorwärts bewegen, rückwärts bewegen, rückwärts drehen, links drehen, rechts drehen

4. Schreibe eine Anleitung dazu: Beginne auf dem Regenbogen, in Richtung des Hundes zeigend. Ende auf dem Vogel.

_____

_____

_____

5. Schreibe eine Wegbeschreibung dazu: Beginne auf dem Regenbogen und zeige auf den Hund. Berühre den Hund NICHT. Berühre NICHT die Katze. Auf dem Vogel enden.

_____

_____

_____

6. Schreibe eine Anleitung dazu: Beginne auf dem Diamanten und zeige auf den Beachball. Benutze KEINE 'vorwärts bewegen'-Befehle. Ende auf dem Beachball.

_____

_____

_____

Name_____

U2-1.1a: Mach ein Sandwich

Weißt du, wie man ein Erdnussbutter-Marmeladen-Sandwich macht? Kannst du es einem Roboter beibringen?

Es gibt viele Dinge, die du so gut beherrschst, dass dir die Aufgabe sehr einfach erscheint. Diese 'leichten Aufgaben' können aber tatsächlich aus vielen einzelnen Schritten bestehen. Das Herstellen eines Erdnussbutter-Marmeladen-Sandwiches ist ein gutes Beispiel dafür. Um das Sandwich zu machen, musst du viele verschiedene Dinge tun, und du musst jedes einzelne in der richtigen Reihenfolge machen, um am Ende ein Sandwich zu erhalten.

Fachjargon

Sequenz bedeutet, der Reihe nach und Schritt für Schritt vorzugehen.

Aufgabe 1: Schreibe Schritt für Schritt auf, wie man ein Sandwich macht.

1. Was musst du tun, um ein Sandwich zu machen? Schreibe jeden Schritt der Reihe nach auf.

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

Name_____

Aufgabe 2: Folge den Anweisungen

Dein Lehrer wird dir helfen, dich auf diese Aufgabe vorzubereiten.

Mach dir ein Sandwich und folge dabei den aufgeschriebenen Anweisungen. Achte darauf, dass du die Anweisungen genauso befolgst, wie sie geschrieben stehen. Füge keine zusätzlichen Schritte hinzu!

2. Wie ist dein Sandwich geworden?

_____

_____

_____

Name_____

U2-1.1b: Menschliche Roboter

Es gibt viele Dinge, von denen du weißt, wie man sie macht, von denen du denkst, dass sie einfach sind, die aber in Wirklichkeit viele Einzelschritte haben. Roboter können einzelne Schritte nicht sehr gut in einen Topf werfen. Stattdessen brauchen Roboter klare Anweisungen, die jede einzelne Bewegung Schritt für Schritt erklären. Um einen Roboter dazu zu bringen, das zu tun, was du willst, ist es wichtig, mit klaren Anweisungen zu erklären, was zu tun ist.

Fachjargon

Sequenz bedeutet, der Reihe nach und Schritt für Schritt vorzugehen.

Aufgabe 1: Schreibe die Schritt-für-Schritt-Anweisungen auf.

Dein Lehrer wird dir helfen, dich auf diese Aufgabe vorzubereiten.

1. Was musst du tun, um den menschlichen Roboter ans Ziel zu bringen? Schreibe jeden Schritt der Reihe nach auf.

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

Aufgabe 2: Folge den Anweisungen

Dein Lehrer wird dir helfen, dich auf diese Aufgabe vorzubereiten. Achte darauf, dass du die Anweisungen genauso befolgst, wie sie geschrieben sind. Füge keine zusätzlichen Schritte hinzu!

2. Hat der menschliche Roboter das Ziel erreicht? Musstest du etwas an deinen Anweisungen ändern, damit die Anweisungen funktionieren?

_____

_____

Name_____

U2-1.2: Gehen wir in EdScratch Schritt für Schritt vor

Wenn du in EdScratch ein Programm für deinen Edison-Roboter schreibst, schreibst du die Anweisungen, die dem Roboter sagen werden, was er tun soll und in welcher Reihenfolge er die einzelnen Dinge tun soll. Jeder EdScratch-Block ist eine Aktion, von der du dem Roboter sagst, dass er sie ausführen soll.

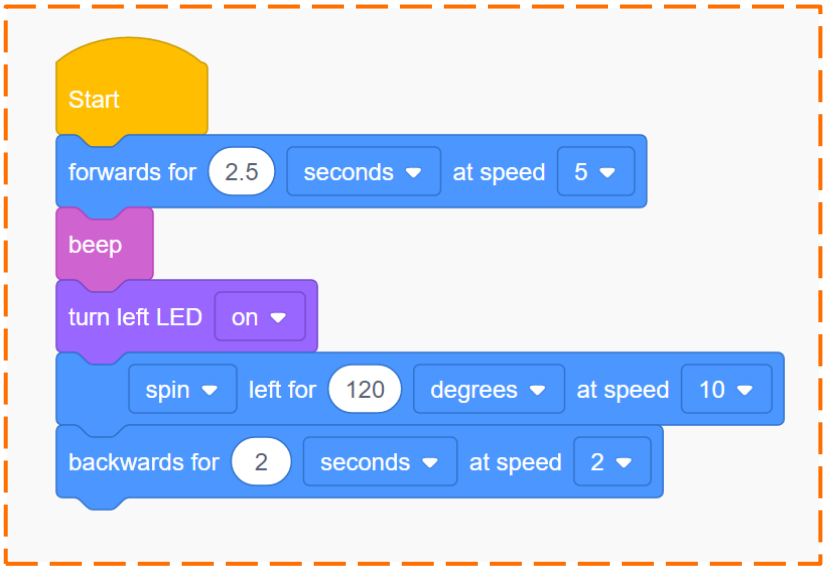
Die Reihenfolge, in der du die Blöcke in deinem Programm miteinander verbindest, sagt dem Roboter, in welcher Reihenfolge er jede Aktion ausführen soll. Edison führt die Aktionen der Reihe nach aus, eine nach der anderen, beginnend mit dem obersten Block.

Nicht vergessen

Sequenz bedeutet, der Reihe nach vorzugehen, Schritt für Schritt.

Aufgabe 1: Was wird Edison tun?

Schau dir dieses EdScratch-Programm an:



Alle EdScratch-Programme müssen den gelben Startblock ganz oben haben. Dies lässt den Roboter wissen, dass das Programm mit dem ersten Block unterhalb des Startblocks startet.

Lies jede Zeile Block für Block, beginnend mit dem obersten Block unterhalb des Startblocks.

1. Wenn du dieses Programm auf Edison herunterlädst, was wird der Roboter tun? Achte darauf, dass du deine Antwort in der richtigen Reihenfolge schreibst.

_____

_____

_____

Name_____

_____

_____

Aufgabe 2: Schreibe das Programm

Gehe zu der EdScratch-App auf deinem Computer und schreibe das Programm vom Bild aus. Finde jeden Block in der Blockpalette und ziehe ihn in den Programmierbereich.

Tipp!

Du kannst die Zahlen in einem Block ändern, indem du auf die Zahl klickst und sie mit deiner Tastatur änderst.

Du kannst einen Dropdown-Eintrag in einem Block ändern, indem du auf den Abwärtspfeil klickst und die gewünschte Option auswählst.

Stelle sicher, dass du alle Blöcke in der richtigen Reihenfolge in dein Programm schreibst. Wenn du dein Programm geschrieben hast, lade es auf deinen Edison-Roboter herunter. Lass das Programm laufen und beobachte, was Edison macht.

Schau dir an, was du in deiner Antwort in Aufgabe eins aufgeschrieben hast, was der Roboter deiner Meinung nach tun würde. Vergleiche, was Edison macht, wenn du das Programm ausführst, mit deiner Antwort. Macht der Roboter das, was du vorhergesagt hast? Wenn nicht, was ist anders? Überprüfe dein Programm und deine Antwort, um zu sehen, ob du den Unterschied erkennen und beheben kannst.

Aufgabe 3: Ändere die Reihenfolge

Versuche die Reihenfolge deines Programms zu ändern. Ändere die Reihenfolge von mindestens drei der Blöcke in deinem Programm. Füge keine weiteren Blöcke hinzu und ändere keine der Optionen innerhalb eines Blocks - ändere nur die Reihenfolge.

Lade dein neues Programm nach Edison herunter und lasse es in deinem Roboter laufen.

2. Welche Blöcke hast du verschoben, um die Reihenfolge des Programms zu ändern? Schreibe dein neues Programm auf.

_____

_____

_____

_____

Name_____

U2-1.3: Lass uns das Fahren mit Edison erforschen

Eine der Block-Gruppen in der EdScratch-Blockpalette ist die Kategorie Drive. Alle Blöcke in dieser Kategorie beziehen sich auf die Verwendung von Edison-Motoren. Eines der Dinge, die du mit den Motoren tun kannst, ist, den Roboter wie ein Auto anzutreiben.

Es wird kein Fahrer benötigt! Nur ein Programmierer.

Wie 'fährt' man einen Edison-Roboter? Indem man den Roboter mit Code programmiert!

Fachjargon

Programme sind die Anweisungen, die du für einen Computer oder einen Edison-Roboter erstellst, um sie zu befolgen. Das Innere eines Programms wird manchmal Code genannt.

Die Leute benutzen oft die Wörter Code und Programm, um dasselbe zu beschreiben. Du könntest zum Beispiel sagen 'schreib ein Programm für Edison' oder 'schreib einen Code für Edison'. In beiden Fällen bedeutet es 'schreibe Befehle, um den Edison-Roboter mit einer Programmiersprache, die er versteht, anzuweisen, was er tun soll'.

Wenn du EdScratch benutzt, um eine Reihe von Anweisungen für Edison zu erstellen, kannst du sagen, dass du programmierst oder dass du codest oder beides. Was dich zu einem Programmierer und einem Coder macht!

Lass uns versuchen, Edison mit Antriebsblöcken zu programmieren.

Name_____

Aufgabe 1: Fahre eine gerade Strecke

Für diese Aufgabe musst du Edison dazu bringen, eine gerade Strecke zu fahren. Benutze dein Arbeitsblatt U2-2. Du musst ein Programm schreiben, damit Edison die Strecke fahren kann. Starte Edison auf dem Umriss und lass den Roboter anhalten, nachdem er die 'Ziellinie' überquert hat.

Rufe die EdScratch-App auf deinem Computer auf. Schau dir die Blöcke in der Kategorie 'Drive' an. Welche Blöcke brauchst du, um dein Programm zu schreiben? Teste dein Programm, indem du es auf deinen Edison-Roboter lädst und es mit Hilfe des Arbeitsblattes ausführst. Hat es funktioniert?

Nicht vergessen

Du kannst die Zahlen in einem Block ändern, indem du auf die Zahl klickst und sie mit deiner Tastatur änderst.

Du kannst einen Dropdown-Eintrag in einem Block ändern, indem du auf den Abwärtspfeil klickst und die gewünschte Option auswählst.

1. Du kannst eine Lösung für diese Aufgabe mit nur einem Block programmieren! Welcher Block wird funktionieren? Fülle den untenstehenden Block so aus, wie du ihn in deinem erfolgreichen Ein-Block-Programm verwendet hast.

Aufgabe 2: Ein Labyrinth fahren

Für diese Aufgabe musst du Edison dazu bringen, durch ein Labyrinth zu fahren. Schau dir das Labyrinth auf dem Arbeitsblatt U2-3 an. Denke über die verschiedenen Aktionen nach, die Edison ausführen muss, um durch das Labyrinth zu fahren. Achte auch auf die Reihenfolge der Aktionen!

2. Was denkst du, was Edison tun muss, um das Labyrinth zu vervollständigen? Schreibe einen Plan auf, um Edison durch das Labyrinth zu bringen.

_____

_____

_____

_____

Name_____

Benutze diesen Plan, um dir zu helfen, ein Programm in EdScratch zu schreiben, mit dem Edison durch das Labyrinth fahren kann. Du wirst mehrere verschiedene Blöcke in deinem Programm verwenden müssen, um das Labyrinth zu vervollständigen. Du musst auch herausfinden, welche Eingabeparameter du in jedem Block verwenden musst.

Fachjargon

Die Dinge, die du innerhalb eines Blocks ändern kannst, wie die Zahlen und Auswahlmöglichkeiten in den Dropdown-Listen, werden als Eingabeparameter bezeichnet.

Starte Edison auf dem Umriss des Labyrinths und lass den Roboter anhalten, nachdem er die 'Ziellinie' überquert hat. Achte darauf, dass Edison während des gesamten Labyrinths innerhalb der Linien bleibt - kein Schummeln!

Tipp

Es kann sein, dass dein Programm beim ersten Mal nicht funktioniert - und das ist okay! Ein Teil des Programmierens ist das Experimentieren und das Lösen von Problemen. Wenn dein Programm nicht funktioniert, denke über die Aktionen nach, die Edison nacheinander ausführen muss, um das Labyrinth zu vervollständigen.

Vermisst du irgendwelche Aktionen in deinem Programm? Gibt es irgendwelche Aktionen, die nicht in Ordnung sind?

Du kannst auch versuchen, einige deiner Eingabeparameter zu ändern. Manchmal macht es einen großen Unterschied, einen Eingabeparameter auch nur ein bisschen zu ändern!

Name_____

U2-1.3a: Irrgarten-Wahnsinn

Hast du Edison dazu gebracht, das Labyrinth auf dem Arbeitsblatt U2-3 erfolgreich zu fahren? Probiere diese anderen Labyrinth-Herausforderungen aus! Denke daran, dass Edison während des gesamten Labyrinths innerhalb der Linien bleiben muss - kein Schummeln!

Spur spiegeln

Vervollständige das Labyrinth, indem du von der 'Ziellinie' startest und zum Start fährst.

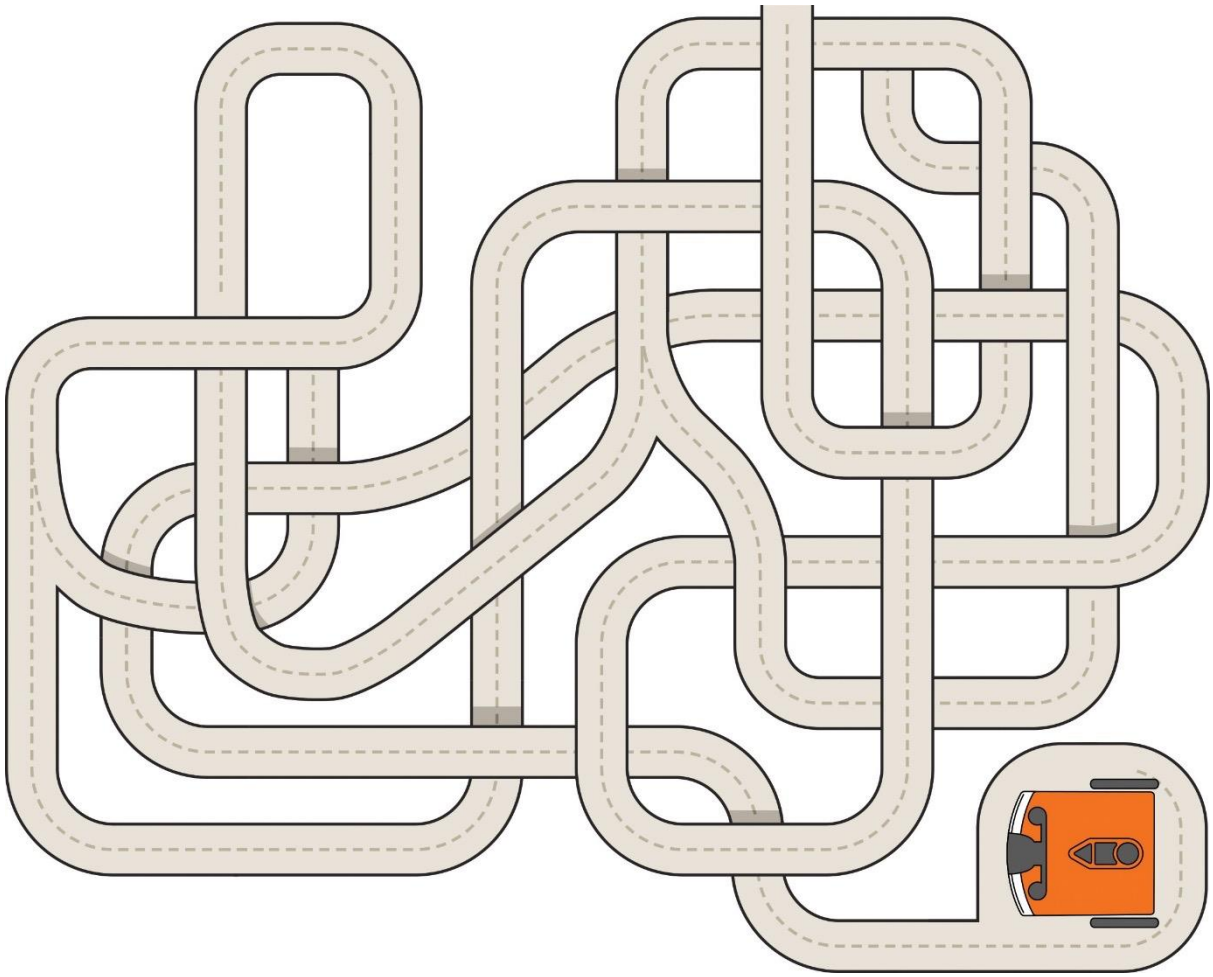
Rückwärts Bot

Vervollständige das Labyrinth, indem du rückwärts vom Anfang des Labyrinths bis zum Ende fährst.

Erstelle dein eigenes Labyrinth

Erstelle dein eigenes Labyrinth, durch das Edison fahren kann, und schreibe dann ein EdScratch-Programm, damit Edison dein Labyrinth vervollständigen kann.

Sobald du dein Labyrinth bewältigt hast, tausche mit einem Partner. Während sie eine Lösung ausarbeiten, um Edison durch dein Labyrinth zu bringen, musst du herausfinden, wie du Edison dazu bringst, durch ihr Labyrinth zu fahren!



Name_____

U2-1.3b: Selbstlaufendes Haustier

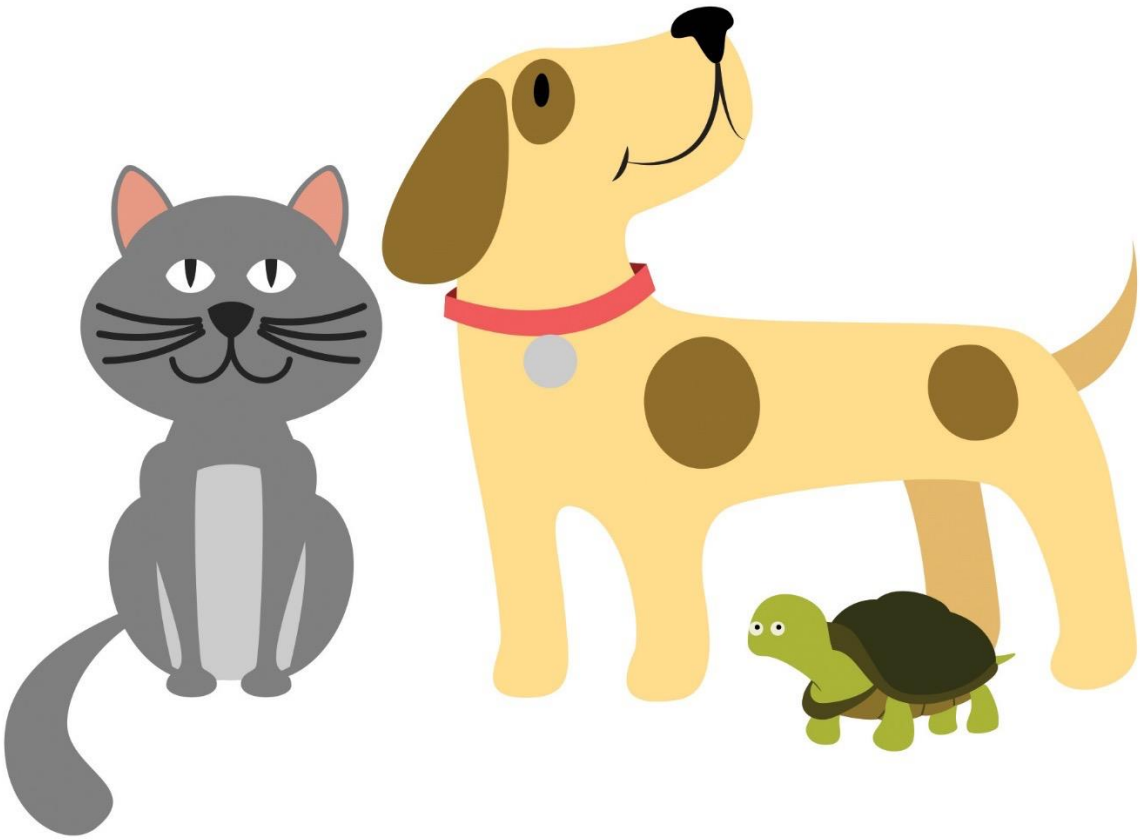
Ein Haustier zu haben bedeutet, sich um dieses Haustier zu kümmern. Du musst es füttern, hinter ihm aufräumen und dafür sorgen, dass es Bewegung bekommt. Wenn es nur einen einfacheren Weg gäbe...

Versuche bei dieser Aktivität Edison in ein Tier zu verwandeln, das sich selbst spazieren führen kann.

Wie kannst du Edison in ein „selbstlaufendes" Haustier verwandeln? Hier sind ein paar Ideen:

- Du könntest Edison dekorieren, um den Roboter in ein Haustier zu verwandeln.
- Du könntest ein Haustier bauen, das an Edison befestigt wird.
- Du könntest etwas bauen, das an der Außenseite von Edison angebracht wird und mit dem Edison herumlaufen kann.

Erstelle dein Haustier. Dann schreibst du ein Programm, damit dein Haustier spazieren geht und zu dir zurückkommt!



Name_____

U2-2.1: Lass uns Edisons Ausgänge erforschen

Was machen Computer? Computer verarbeiten Informationen. Das bedeutet, dass Computer Informationen von irgendwo nehmen und etwas mit diesen Informationen machen. Zum Beispiel kannst du einem Computer zwei Zahlen geben und ihm sagen, er soll sie zusammenzählen. Der Computer kann dann diese Zahlen addieren und dir das Ergebnis zeigen.

Dieser Zyklus von Informationen, die hereinkommen, wobei der Computer etwas mit den Informationen macht und dann irgendein Ergebnis erzeugt, wird als EVA-Prinzip bezeichnet.

Fachjargon

Eingaben (Inputs) sind die Informationen und Anweisungen, die du einem Computer gibst.

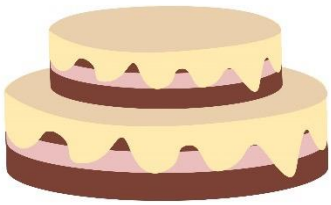
Eine Verarbeitung ist das, was der Computer mit einem Computerprogramm voller Informationen und Anweisungen macht. Dies wird manchmal als 'Ausführen' des Programms bezeichnet.

Ausgaben (Outputs) sind die Ergebnisse, die du von einem Computer erhältst. Was der Computer anzeigt, oder wie sich der Roboter verhält, sind die Ausgaben, die du aufgrund der Informationen und Anweisungen bekommst, die du dem Computer gegeben hast.

Wir nennen diesen Prozess der Eingaben, der Verarbeitung von Informationen und die Erzeugung einer Art von Ausgabe das EVA-Prinzip.

Das EVA-Prinzip wird nicht nur bei Computern verwendet. Du kannst diesen Zyklus auch in deinem täglichen Leben in Aktion sehen.

Das Backen eines Kuchens ist ein gutes Beispiel. Du gibst Zutaten in eine Form und schiebst diese Form in den Ofen. Der Prozess des Backens findet dann im Ofen statt. Nach einer Weile ist der Kuchen fertig!



Name_____

Eingaben, Ausgaben und Edison

Wenn du ein Programm für deinen Edison-Roboter schreibst, sagst du dem Roboter, was er tun soll, indem du ihm Eingaben machst. Edisons Mikrochip verarbeitet dann die Informationen, um dem Roboter zu sagen, was er ausgeben soll.

Dein Edison-Roboter hat drei Haupttypen von Outputs: Outputs durch die Motoren, Outputs durch die LEDs und Outputs durch Geräusche. In EdScratch sind die Blöcke, die mit Edisons Hauptausgängen zu tun haben, in drei verschiedene Kategorien eingeteilt: Drive, LEDs und Sound.

1. Schau dir die Blöcke in den Kategorien Drive, LEDs und Sound in EdScratch an. Welche Kategorie enthält die Code-Blöcke, die du in ein Programm eingeben müsstest, damit Edison jeden Output generiert? Ordne jede Ausgabe der Kategorie zu, in der du die Blöcke findest, die du benötigen würdest.

Ausgabe	Kategorie
Edisons Lichter ausschalten ☐	● Drive
Spiele eine Melodie. ☐	● LEDs
Edison rechts drehen ☐	● Sound

Name_____

Aufgabe 1: Blinken und Piepen

Edison hat zwei rote LED-Leuchten. Wenn du Edison einschaltest, kannst du sehen, wie diese beiden LEDs zu blinken beginnen.

Edison hat auch eine besondere Technik, die man links neben dem runden Knopf oben auf dem Roboter sehen kann. Das ist ein Buzzer und ein Geräuschsensor in einem. Er kann Geräusche erkennen, aber er kann auch Lärm machen!

Für diese Aufgabe müsst ihr ein Programm schreiben, das Edison dazu bringt, sowohl die LED als auch die Summerausgänge zu benutzen. Schreibe in EdScratch ein Programm mit acht Blöcken, das Edison anweist, die folgenden Dinge der Reihe nach zu tun:

- ☐ schalte das linke LED-Licht ein
- ☐ piep
- ☐ schalte das linke LED-Licht aus
- ☐ piep
- ☐ mach das rechte LED-Licht an
- ☐ piep
- ☐ schalte das rechte LED-Licht aus
- ☐ piep

Lade dein Programm herunter und teste es mit deinem Edison-Roboter.

2. Hat das Programm so funktioniert, wie du erwartet hast? Konntest du beobachten, wie der Roboter jeden Schritt des Programms ausführt?

_____

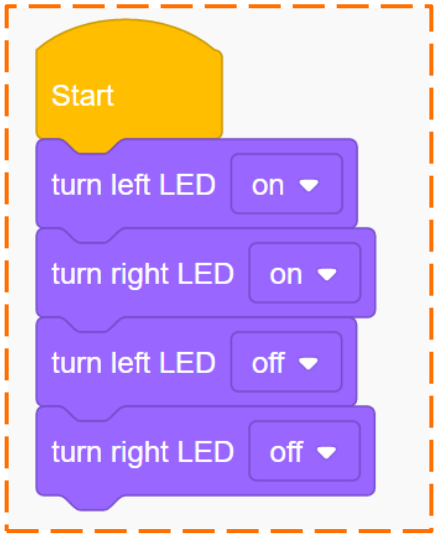
_____

_____

Name_____

Aufgabe 2: Edison zum Blinzeln bringen

Einige von Edisons Ausgaben, wie das Ein- und Ausschalten einer LED, geschehen sehr schnell. Tatsächlich kann das so schnell passieren, dass es wirklich schwer zu sehen ist.



Versuche das folgende Programm in EdScratch zu schreiben:

Lade es herunter und starte es mit deinem Roboter. Kannst du Edison blinken sehen?

Weil der Roboter seine LEDs blinken lässt, wenn er sich im Standby-Modus befindet, wird es wirklich schwer sein, dieses Programm in Aktion zu sehen.

Woran liegt das?

Edison bewegt sich nacheinander durch jeden EdScratch-Block, aber der Roboter kann jeden Block sehr schnell bearbeiten. Computer können Informationen sehr schnell verarbeiten - das ist eines der Dinge, die sie so nützlich machen!

Wenn du willst, dass Edison eine Pause macht, nachdem er einen Block beendet hat, bevor er zum nächsten Block weitergeht, musst du ihm das sagen.

Eine der EdScratch-Blockkategorien ist die Kategorie Control. Die Blöcke in der Kategorie Control ermöglichen es dir, den Ablauf deines Programms zu kontrollieren. Einer der Blöcke in dieser Kategorie ist der Warte (wait)-Block:

Name_____

Dieser Block sagt Edison, dass er die von dir angegebene Zeitspanne warten soll, bevor er zum nächsten Block weitergeht. Lass uns versuchen, diesen Block in einem Programm zu verwenden.

Ändere das 'blinkende' Programm von vorher, aber benutze diesen neuen Kontrollblock, damit das Programm besser funktioniert. Du willst ein Programm, in dem es sehr einfach ist, Edison 'blinken' zu sehen. Experimentiere mit mindestens einem Warteblock. Teste mit Warteblöcken an verschiedenen Stellen in deinem Programm, um zu sehen, was am besten funktioniert.

3. Wie sieht dein Programm aus? Welche Blöcke benutzt es, in welcher Reihenfolge? Schreibe dein Programm unten auf. Achte darauf, dass du die von dir verwendeten Eingabeparameter angibst.

Mini-Herausforderung!

Was passiert, wenn eine Person blinkt? Ihre Augen beginnen sich zu öffnen und dann...

Schau dir dein Blinzelprogramm an. Kannst du dein Programm so einstellen, dass es mehr wie ein Blinken aussieht?

Name_____

U2-2.1a: Fahre sicher durch das Labyrinth

Du kannst Programme für Edison schreiben, die deinem Roboter sagen, dass er mehrere Arten von Ausgaben verwenden soll.

Nicht vergessen

Dein Edison-Roboter hat drei Haupttypen von Ausgängen: Ausgänge mit den Motoren, Ausgänge mit den LEDs und Ausgänge mit Sounds. In EdScratch sind die Blöcke, die mit Edison's Hauptausgängen zu tun haben, in drei verschiedene Kategorien eingeteilt: Antrieb, LEDs und Sound.

Für diese Aktivität musst du ein Programm schreiben, das Edison sagt, dass er das Labyrinth auf dem Arbeitsblatt U2-3 fahren soll. Dieses Mal muss Edison jedoch ein sehr sicherer Fahrer sein. Auf der Straße benutzen die Fahrer ihre Blinklichter und die Hupe, um andere Fahrer zu warnen. Edison kann diese Dinge auch tun!

Fahre das Labyrinth, indem du an der Außenlinie beginnst und vorwärts zur Ziellinie fährst.

Dein Programm sollte enden, nachdem Edison die Ziellinie überquert hat. Dein Programm muss Edison sagen, dass er eine Pause machen und die LED-Leuchten 'anzeigen' soll, bevor er jede Kurve im Labyrinth fährt. Stelle sicher, dass andere Fahrer die LED-Anzeigen sehen können!

Tipp!

Stell dir vor, der 'Piep'-Block ist wie eine Hupe in einem Auto. Wo in deinem Programm könnte es Sinn machen, dass Edison piepst?

Dein Programm sollte auch den 'Piep'-Block mindestens einmal irgendwo im Programm benutzen.

Name_____

Tipp!

Wenn du nach links abbiegst, welche LED solltest du benutzen, um anzuzeigen? Was ist, wenn du rechts abbiegst?

Name_____

U2-2.2: Lass uns die Eingabeparameter untersuchen

Jeder Block in EdScratch enthält Eingaben, die Edison mitteilen, was der Roboter tun soll. Du hast wahrscheinlich bemerkt, dass viele der Blöcke auch Eingabeparameter haben.

Fachjargon

Eingabeparameter sind die Dinge, die du innerhalb eines Blocks ändern kannst, wie zum Beispiel die Zahlen und Auswahlmöglichkeiten in den Dropdown-Listen. Du kannst dir Eingabeparameter als spezifische Informationen vorstellen, die in einem Input benötigt werden.

Wenn du z.B. willst, dass Edison vorwärts fährt, musst du dem Roboter spezifische Informationen über diesen Befehl geben, z.B. wie weit er fahren soll und mit welcher Geschwindigkeit.

Es gibt drei Arten von Eingabeparametern in EdScratch:

- **Zahlen**, die du mit deiner Tastatur in den Block tippst,
- **Drop-Down-Menüs**, in denen du eine Option auswählst und
- **runde oder rautenförmige Löcher**, in die man spezielle Blöcke einfügt.

Jeder Eingabeparameter in einem Block gibt eine andere Information, die Edison benötigt, um diesen Befehl ausführen zu können. Du kannst dir Eingabeparameter als die Antworten auf Fragen vorstellen, die der Roboter zu dem hat, was du von ihm verlangst. Wenn du zum Beispiel willst, dass Edison rückwärts fährt, musst du drei Fragen beantworten:

Name_____

1. **Frage**: Wie weit soll der Roboter fahren?

   →Antwort: Distanz-Eingabeparameter

2. **Frage**: Welche Einheiten benutzt du, um die Entfernung zu messen?

   →Antwort: Entfernungseinheiten Eingabeparameter

3. **Frage**: Wie schnell soll der Roboter fahren?

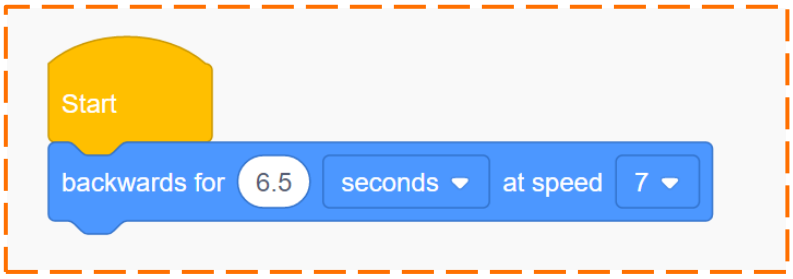
   →Antwort: Geschwindigkeits-Eingabeparameter

Du musst immer alle Eingabeparameter in einem Block ausfüllen, um dem Roboter alle Informationen zu geben, die er braucht.

Probier es aus!

Die Eingabeparameter in einem Programm geben dir viele Informationen darüber, was dieses Programm dem Edison-Roboter abverlangt. Versuche die folgenden Programme zu lesen und beantworte die Fragen.

Schau dir dieses Programm an:



1. Was ist in diesem Programm der Eingabeparameter für Entfernungseinheiten?

   _____

2. Was ist in diesem Programm der Eingabeparameter für die Geschwindigkeit?

   _____

3. Was ist das volle Programm, das Edison tun soll? Markiere alle Eingabeparameter in deiner Antwort in einer anderen Farbe oder durch Unterstreichen oder Einkreisen der einzelnen Parameter.

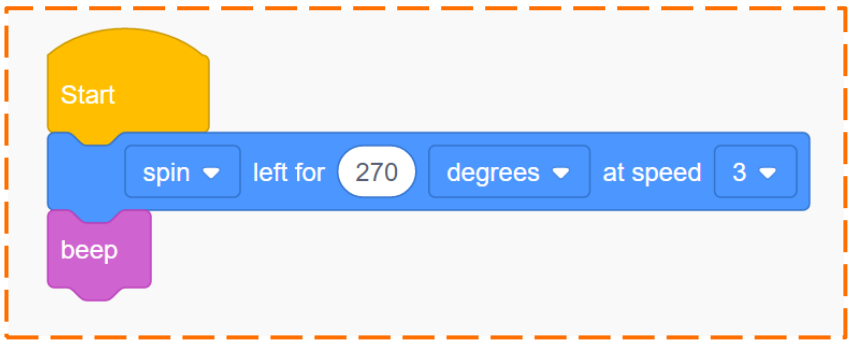
   _____

   _____

   _____

Name_____

Schau dir dieses Programm an:



4. Wie viele Eingabeparameter gibt es in diesem Programm?

   _____

5. Denke daran, dass Eingabeparameter Antworten auf Fragen sind, die der Roboter wissen muss. Auf welche Frage antwortet der Spin-Eingabeparameter? *Tipp*: du solltest dir vielleicht die anderen Optionen für diesen Eingabeparameter in EdScratch ansehen, damit du über deine Antwort nachdenken kannst.

   _____

   _____

Name_____

U2-2.2a: Bring Edison bei, bis 9 zu zählen

Hast du dir jemals angesehen, wie alte digitale Schilder und Uhren Zahlen anzeigen? Die Zahlen werden durch Linien dargestellt, die ein rechteckiges Gitter bilden. Jede Zahl, von 0 bis 9, kann durch eine Kombination der Linien in diesem Gitter angezeigt werden.

Zahlen auf Digitalanzeigen haben keine Kurven oder diagonalen Linien. Sie haben nur gerade Linien und rechte Winkel. Das macht sie zu perfekten Mustern, mit denen Edison fahren kann!

Aufgabe 1: Bring Edison eine Zahl bei

Schau dir die Aktivitätsblätter U2-4, U2-5, U2-6 und U2-7 an. Wähle eines der Aktivitätsblätter aus.

Schreibe ein Programm für Edison, so dass der Roboter über die von dir gewählte Nummer der Digitalanzeige verfolgen kann. Starte den Roboter über die Nummer der Digitalanzeige und fahre so, dass der Roboter über jedes Segment der Nummer fährt.

1. Welche Nummer der Digitalanzeige hast du benutzt?

   _____

2. Wie sieht dein Programm aus? Welche Blöcke benutzt es, in welcher Reihenfolge? Schreibe dein Programm unten auf. Achte darauf, dass du die von dir verwendeten Eingabeparameter angibst.

Name_____

3. Achte auf den Eingabeparameter distance in den Blöcken in deinem Programm. Was fällt dir an den von dir verwendeten Eingaben auf? Wie könnte dir das helfen, ein Programm für eine andere Digitalanzeigenummer zu planen?

_____

_____

_____

Aufgabe 2: Bring Edison eine andere Zahl bei

Wähle ein anderes Arbeitsblatt mit Digitalanzeige. Benutze das, was du über den Entfernungs-Eingabeparameter von deiner letzten Nummer gelernt hast und schreibe ein Programm für Edison, damit der Roboter deine neue Digitalanzeigenummer zurückverfolgen kann.

4. Welche Nummer der Digitalanzeige hast du benutzt?

   _____

5. Wie sieht dein Programm aus? Welche Blöcke benutzt es, in welcher Reihenfolge? Schreibe dein Programm unten auf. Achte darauf, dass du die von dir verwendeten Eingabeparameter angibst.

Name_____

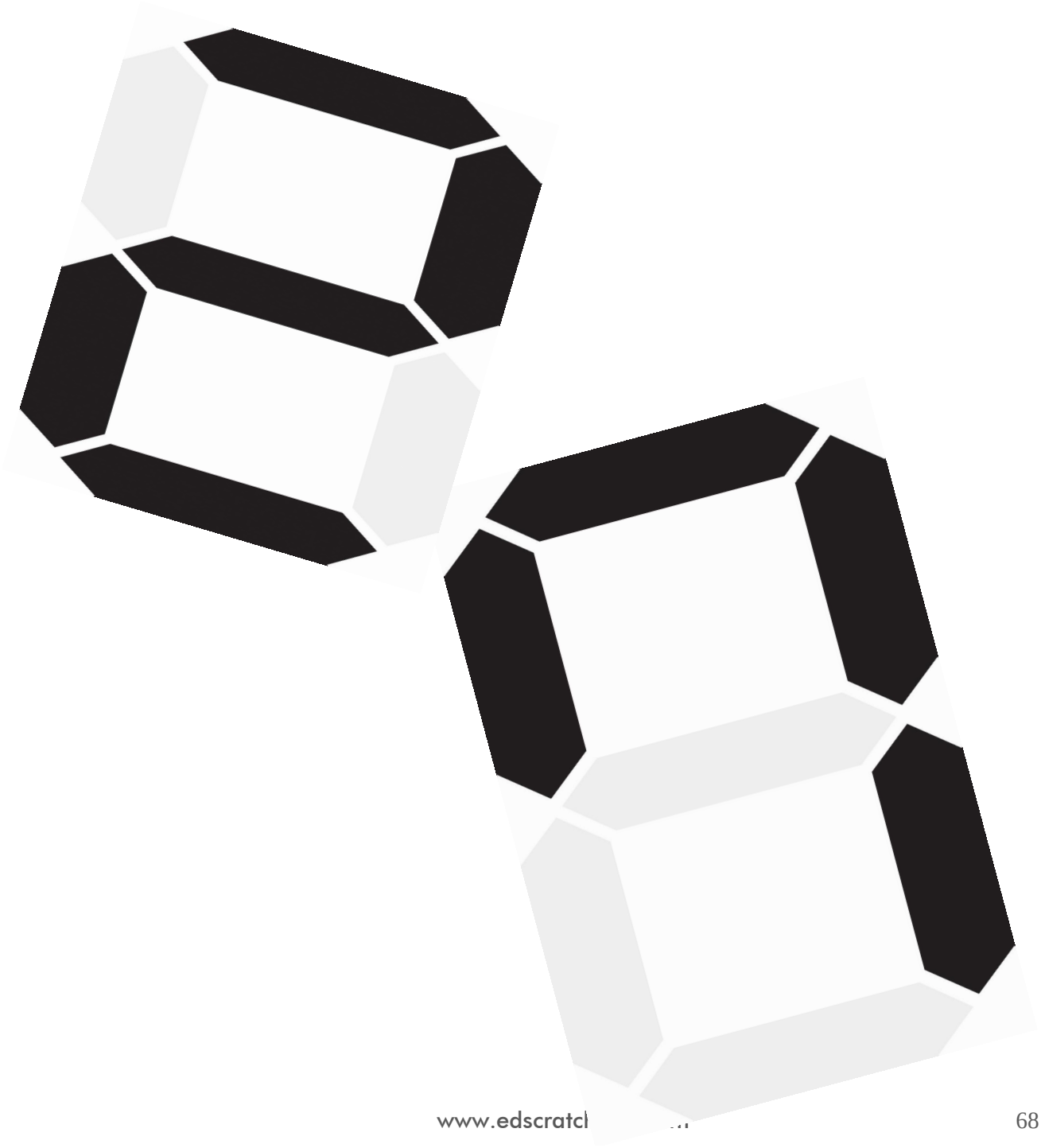
6. Vergleiche deine beiden Programme. Fällt dir ein Muster auf, das in beiden ähnlich ist? Welche sind das?

_____

_____

_____

_____



Name_____

U2-2.2b: Bring Edison bei, laut bis 9 zu zählen

Kannst du Edison dazu bringen, die Nummer einer Digitalanzeige zu verfolgen, indem du darüber fährst und denselben Wert 'laut' in nur einem Programm zählst?

Was zu tun ist

Schau dir die Aktivitätsblätter U2-4, U2-5, U2-6 und U2-7 an. Wähle eines der Aktivitätsblätter aus.

Schreibe ein Programm für Edison, so dass der Roboter über die von dir gewählte Nummer der Digitalanzeige fährt.
Du brauchst Edison auch, um irgendwie 'laut' zu zählen.

Dein Programm muss Edison die gleiche Menge zählen lassen wie die Nummer der Digitalanzeige, die er fährt. Wenn du zum Beispiel die Zahl 5 wählst, muss dein Programm Edison eine Art Signal geben, wenn es bis 5 'zählt'.

Tipp!

Welchen von Edisons Ausgängen könntest du benutzen, um dem Roboter zu signalisieren, dass er zählt?

Denke über die Reihenfolge der Dinge nach, die Edison tun soll. Wird der Roboter den ganzen Weg fahren und dann zählen? Zählt er vor dem Fahren? Ein bisschen fahren, bis eins zählen, dann noch ein bisschen mehr fahren, bevor er die nächste Zahl zählt?

Wie du das machst, liegt ganz bei dir!

Name_____

Mini-Herausforderung!

Was ist mit den restlichen Zahlen? Erstelle deine eigene Nummer für die Digitalanzeige mit einer anderen Nummer als auf den Arbeitsblättern.

Nicht vergessen

Zahlen auf Digitalanzeigen haben keine Kurven oder diagonale Linien. Sie haben nur gerade Linien und rechte Winkel. Die Zahlen werden durch Segmente dargestellt, die ein rechteckiges Gitter bilden. Jede Zahl, von 0 bis 9, kann durch eine Kombination der Linien im Gitter angezeigt werden.

Wenn du deine digitale Nummer gemacht hast, teste sie aus! Schreibe ein Programm, mit dem Edison deine Nummer verfolgen und 'zählen' kann.

Name_____

U2-2.3: Lass uns Edisons musikalische Talente erkunden

Outputs, die Sounds verwenden, sind eine der drei Hauptarten von Outputs deines Edison-Roboters. In EdScratch befinden sich die Blöcke, die sich auf Soundausgaben beziehen, in der Kategorie Sound.

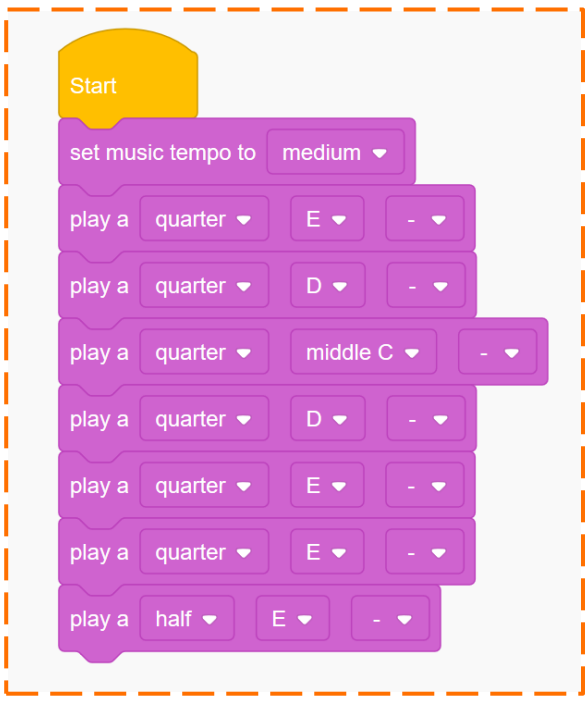
Nicht vergessen

Edison hat ein besonderes Stück Technik, das ihr links neben dem runden Knopf sehen könnt. Dies ist ein Buzzer und ein Schallsensor in einem. Er kann Geräusche erkennen, aber er kann auch Geräusche machen, wie Piepstöne oder Musiknoten.

Wirf einen Blick auf die Kategorie Sound in EdScratch. Es gibt nicht viele Blöcke in dieser Kategorie, also könnte man meinen, dass man mit Edison und Sounds nicht viel anfangen kann. Indem du die Sound-Blöcke in unterschiedlicher Reihenfolge und mit unterschiedlichen Eingabeparametern verwendest, kannst du jedoch ganze Lieder abspielen!

Aufgabe 1: Eine Melodie spielen

Schreibe in EdScratch das folgende Programm:



Name_____

Lade das Programm auf deinen Edison-Roboter herunter und führe es aus. Dieses Programm ist der erste Teil eines Songs, den du vielleicht kennst. Erkennst du die Melodie wieder?

Der erste Block im Programm ist der Block mit dem eingestellten Musiktempo (set music tempo). Klicke auf das Drop-Down-Menü im set-music-tempo-Block, um die Optionen der Eingabeparameter für den Block zu sehen. Wähle einen anderen Eingabeparameter für den Block und lade dann das angepasste Programm nach Edison herunter. Starte dein neues Programm in deinem Roboter.

1. Welchen neuen Eingabeparameter hast du für den eingestellten Musik-Tempo-Block gewählt?

_____

2. Was ist passiert, als du das neue Programm gespielt hast? Was hat sich im Vergleich zum ursprünglichen Programm geändert?

_____

_____

_____

3. Was bewirkt der Tempo-Block der eingestellten Musik?

_____

_____

_____

4. Wenn du den eingestellten Musiktempoblock am Ende dieses Programms statt am Anfang setzen würdest, was würde passieren? Tipp: Denke daran, dass Edison jeden EdScratch-Block einzeln nacheinander vom Anfang des Programms abliest.

_____

_____

_____

Name_____

Tipp!

Nicht sicher, was passieren würde, wenn du eine Änderung an einem Programm vornimmst? Einer der besten Wege, das herauszufinden ist, die Änderung vorzunehmen, das angepasste Programm herunterzuladen und es mit Edison auszuprobieren!

Du kannst jederzeit mit dem Programmieren experimentieren!

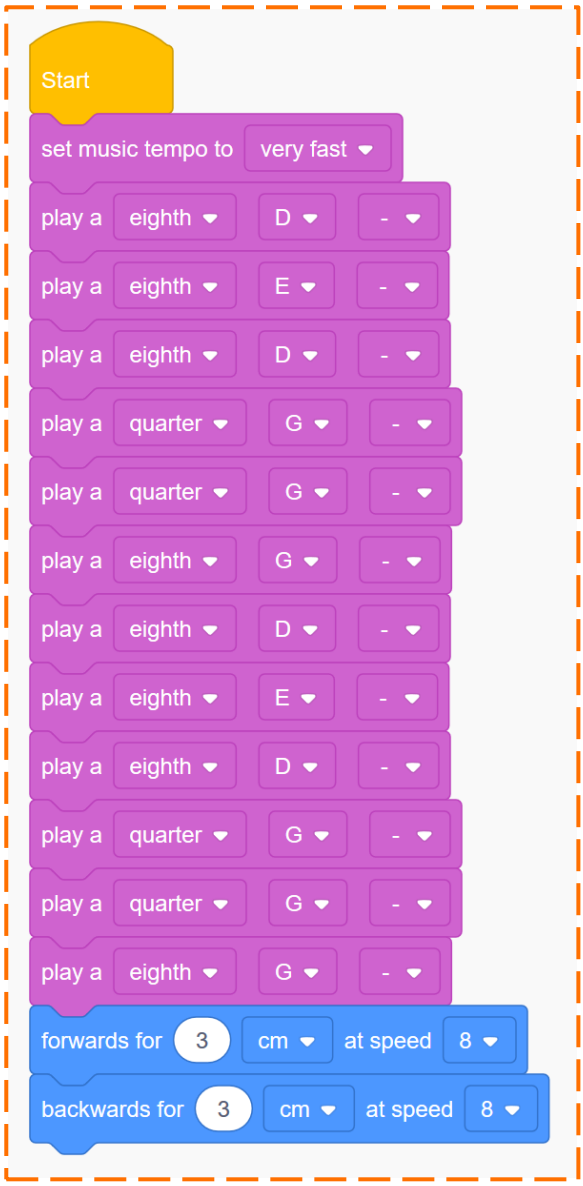
Benutze diesen Link

Willst du Edison mehr von diesem Lied spielen hören?
Öffnen Sie das Programm über diesen Link:
https://www.edscratchapp.com?share=Eb12x3Dm

Name_____

Aufgabe 2: Zur Musik bewegen

Schau dir dieses EdScratch Programm an:



Dieses Programm hat die Musik für den ersten Teil des Songs The Hokey-Pokey. Das ist ein toller Song zum Mittanzen - der Song sagt dir genau, welche Bewegungen du machen musst!

Schreibe in EdScratch das Hokey-Pokey-Tanzprogramm. Lade das Programm auf deinen Edison-Roboter herunter und spiele es ab.

5. Hat sich dein Edison-Roboter mit der Musik mitbewegt? Warum glaubst du, dass dies der Fall ist?

_____

_____

_____

Name_____

Edison kann eine Musiknote spielen, während er etwas anderes macht, wie z.B. Autofahren, aber du musst dem Roboter sagen, dass es das ist, was du willst, dass er tut. Es gibt einen speziellen Block in der Kategorie Sound in EdScratch, den du dafür benutzen musst. So sieht er aus:

Der Block „Musik im Hintergrund abspielen“(play music in background) ist ein Gruppierungsblock. Andere Blöcke können in diesem Gruppierungsblock sitzen.

Woran liegt das?

Die Form eines Blocks in EdScratch kann dir einen Hinweis darauf geben, wofür dieser Block in Programmen verwendet wird. Schau dir den Block play music in background an. Siehst du, dass er eine Form hat, die ein bisschen wie ein Krokodilmaul aussieht? Andere Blöcke können in der Öffnung des ‚Krokodilmauls' dieses Blocks sitzen.

Jeder Block, der sich innerhalb des „play music in background“-Block befindet, wird von diesem Gruppierungsblock beeinflusst. Denke daran, Edison wird jedem EdScratch-Block einzeln folgen. Der Roboter sieht den Gruppierungsblock zuerst und weiß, dass alle Blöcke innerhalb dieses Blocks den ‚play music in background'-Effekt erhalten.

Füge einen „Musik im Hintergrund abspielen“(play music in background)-Block zu deinem Hokey-Pokey-Tanzprogramm hinzu. Denke darüber nach, wo im Programm dieser Block hin muss.

Lade das angepasste Programm auf deinen Edison-Roboter herunter und spiele es ab. Edison sollte jetzt anfangen, das Lied zu spielen und sich gleichzeitig bewegen!

Edison's Tanzbewegungen brauchen allerdings noch ein bisschen Arbeit.

Name_____

Tipp!

Denke daran, dass, wenn etwas in einem Programm nicht ganz richtig ist, eine Warnmeldung in der Bug-Box erscheint. Diese Nachrichten können dir helfen herauszufinden, was innerhalb des Gruppierungsblocks sein sollte und was nicht!

Aufgabe 3: Tanze mit

Füge weitere Tanzschritte zu deinem Hokey-Pokey-Tanzprogramm hinzu. Du könntest Edison dazu bringen, den Schritten in den Hokey-Pokey-Liedtexten zu folgen oder deinen eigenen Tanz zu erfinden!

Kannst du Edison dazu bringen, rechtzeitig zur Musik zu tanzen?

Name_____

U2-2.3a: Spiel ein Lied in einer Runde

Für diese Aktivität musst du mehrere Edison-Roboter zusammen benutzen, um ein Lied in einer Runde zu spielen.

Woran liegt das?

Eine Runde ist ein Musikstück, bei dem zwei oder mehr Leute (oder Roboter!) genau die gleiche Melodie singen oder spielen, aber jede beginnt zu einer anderen Zeit. Während jeder Teilnehmer einen anderen Teil des Liedes singt oder spielt, harmonisiert die Melodie sie dennoch miteinander!

Arbeite in einer Gruppe, um deine Edison-Roboter in harmonisierende Musikstars zu verwandeln!

Was ihr machen müsst

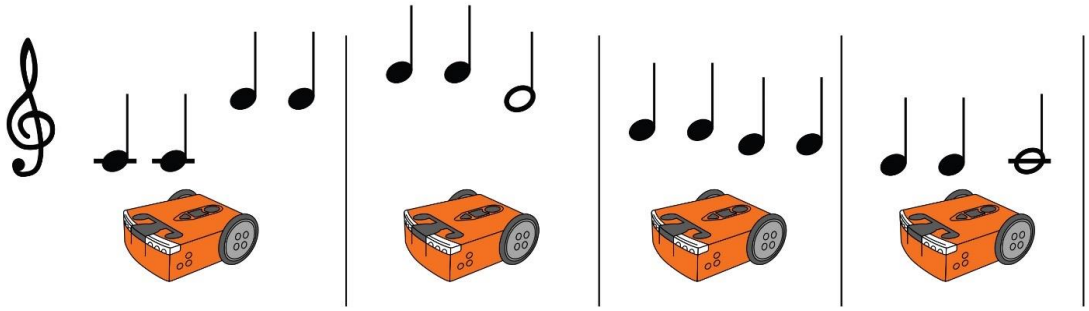
Als erstes musst du einen Song auswählen, den die Roboter spielen sollen. Viele Kinderlieder funktionieren gut in einer Runde. Ein Beispiel ist das Lied Row, Row, Row Your Boat.

Sobald du dein Lied ausgewählt hast, musst du jeden Roboter so programmieren, dass er es spielt. Das Programm jedes Roboters wird ein bisschen anders sein, da jeder Roboter warten muss, bis die richtige Zeit gekommen ist, um in die Runde einzusteigen.

Teste deine Programme mit deinen Robotern. Dann veranstalte eine musikalische Show!

Tipp!

Welcher Kontrollblock sagt Edison, dass er warten soll?



Name_____

U2-2.3b: Du bist der Dirigent

Was ist dein Lieblingslied? Gibt es ein Titellied zu einer TV-Show oder einem Film, das du gerne hörst?

Für diese Aktivität ist die Songauswahl dir überlassen!

Was zu tun ist

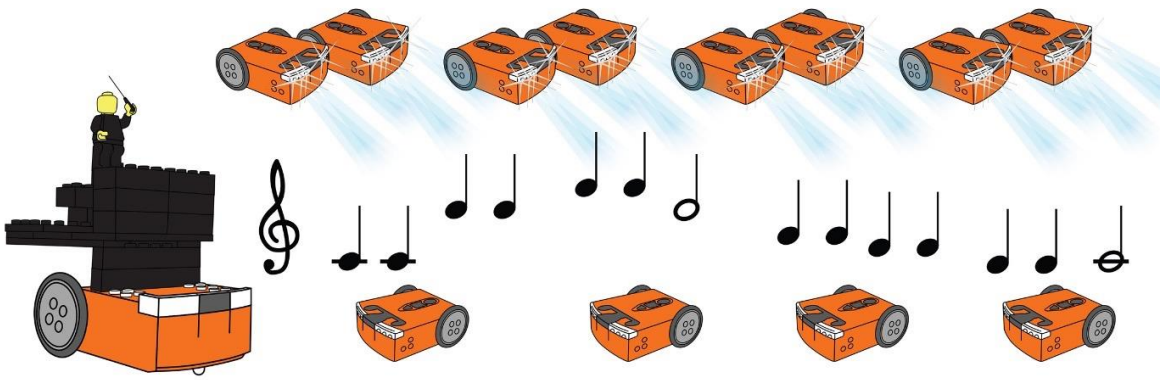
Du bist der Dirigent, und du musst Edison dazu bringen, das Lied deiner Wahl zu spielen. Schreibe ein Programm für Edison, damit der Roboter dein Lied spielt. Teste verschiedene Tempi aus, um zu sehen, welches am besten funktioniert.

Wenn du willst, kannst du den Roboter auch so programmieren, dass er sich zu deinem Lied bewegt oder tanzt.

Wenn du dein Programm fertig hast, lass es in Edison laufen, damit der Roboter dein musikalisches Meisterwerk spielt!

Tipp!

Welcher Block sagt Edison, dass er im Hintergrund Musik spielen soll?



Name_____

U2-2.4: Lass uns Bugs und Fehlerbehebung erforschen

Willst du ein Geheimnis über die Arbeit mit Computern und das Schreiben von Code wissen? Hier ist es:

IRGENDWAS GEHT IMMER SCHIEF!

Okay, das ist also eigentlich kein Geheimnis, aber es ist wirklich wichtig. Ein wichtiger Teil des rechnerischen Denkens und der Arbeit mit Computern ist das Lösen von Problemen.

Nicht vergessen

Computational thinking bedeutet, über ein Problem oder eine Aufgabe auf eine Art und Weise nachzudenken, die der Denkweise eines Computers ähnelt. Es ist eine Art und Weise, Informationen zu planen, Probleme zu lösen und zu analysieren, ähnlich wie es ein Computer tut.

Wenn die Dinge nicht so funktionieren, wie du willst, denk daran, dass das in Ordnung ist! Es bedeutet nur, dass es Zeit ist, das Problem zu lösen. Eine der Hauptarten der Problemlösung, die du in der Informatik machst, ist das Finden und Beheben von Fehlern.

Das Debugging ist ein wichtiger Teil des Programmierens und der Benutzung von Robotern. Leute, die als Informatiker, Computerprogrammierer oder Robotiker professionell arbeiten, verbringen viel Zeit mit dem Debugging. Wahrscheinlich sogar mehr Zeit, als sie ihren Code schreiben!

Fachjargon

Wenn etwas in einem Computerprogramm nicht funktioniert, nennt man das einen Bug.

Das Finden und Beheben von Fehlern in einem Computerprogramm wird Debugging genannt.

Name_____

Woran liegt das?

Etwas, das schief läuft, als ‚Bug' (= engl. Käfer) zu bezeichnen, mag ein bisschen seltsam erscheinen, aber das ist es, wie diese Fehler wirklich genannt werden.

Warum werden Computerprobleme Bugs genannt? Eine Frau namens Grace Murray Hopper, die eine der Erfinderinnen der modernen Computerprogrammierung ist, hat sich diesen Begriff ausgedacht. Sie entdeckte einmal, dass das Problem, das zu einer Fehlfunktion ihres Computers führte, eine echte Motte war, die in die Hardware eingedrungen war! Sie beseitigte das Problem, indem sie ihren Computer buchstäblich debuggte.

Der Name blieb hängen, und jetzt werden Probleme mit Software oder Hardware in Computern als Bugs bezeichnet und ihre Behebung heißt Debugging!

Fehlersuche in EdScratch

Die Bug-Box in der EdScratch-Programmierumgebung ist eine spezielle Funktion, die dir hilft, alle Fehler in deinem Code zu finden und zu beheben.

Nicht vergessen

Die Bug-Box ist der Bereich unter der Blockpalette und dem Programmierbereich in EdScratch. Wenn es einen Fehler gibt oder wenn es den Anschein hat, dass etwas in einem Programm nicht ganz stimmt, wird eine Warnmeldung in der Bug-Box angezeigt.

Die Warnmeldungen, die in der Bug-Box erscheinen, sind dazu da, dir Informationen über jegliche Probleme oder mögliche Probleme in deinem Code zu geben. Diese Nachrichten sind EdScratch's Weg zu sagen, dass es einen Fehler in deinem Code gibt, oder, dass du einen Fehler finden könntest, wenn du dein Programm in Edison ausführst. Im Code gibt es zwei Haupttypen von Fehlern: Syntaxfehler und logische Fehler.

Name_____

Fachjargon

Syntax sind die Regeln, wie eine Programmiersprache funktioniert. Alle Sprachen haben Regeln. Für Sprachen, die die Leute sprechen, wie Englisch oder Chinesisch, gibt es Regeln über Rechtschreibung und Grammatik und wie man die Buchstaben oder Zeichen in dieser Sprache schreibt. Die Syntax ist dasselbe, aber für eine Computersprache.

Syntaxfehler werden durch Probleme in der Art und Weise, wie du deinen Code geschrieben hast, verursacht, die gegen die Regeln der Sprache verstoßen. Diese Fehler sind so etwas wie Tipp- oder Rechtschreibfehler beim Schreiben.

Logik ist eine organisierte Denkweise, die für einen Computer Sinn macht. Die Logik bestimmt den Ablauf eines Programms, wie du die Dinge innerhalb eines Programms anordnest und welche Eingaben du verwendest, um die gewünschten Ausgaben zu erzeugen.

Logische Fehler sind Probleme mit der Logik eines Programms. Logische Fehler sind grundsätzlich Probleme mit der Denkweise in einem Programm. Programme, die für den Computer keinen Sinn ergeben, und Programme, die einen Computer bitten, etwas zu tun, was er nicht tun kann, sind Beispiele für logische Fehler. Wenn ein Programm anders funktioniert, als du erwartet hast, besteht eine gute Chance, dass es irgendwo einen logischen Fehler gibt.

Zu verstehen, ob ein Fehler das Ergebnis eines Syntaxfehlers oder eines logischen Fehlers ist, kann dir helfen, das Problem zu beheben. Wenn es einen Syntaxfehler in deinem EdScratch-Programm gibt, bekommst du eine rote Warnmeldung in der Bug-Box und du kannst das Programm nicht nach Edison herunterladen. Wenn du dein Programm herunterladen und in Edison ausführen kannst, es aber nicht das tut, was du willst, bedeutet das wahrscheinlich, dass ein logischer Fehler vorliegt.

Name_____

Probiere es aus!

Schau dir das folgende Programm an:

Dieses Programm ist voller Bugs. Deine Aufgabe ist es, sie zu beheben!

Tipp!

Einer der besten Wege, um zu sehen, was mit einem Programm falsch - und richtig - läuft, ist, es mit EdScratch und Edison auszuprobieren!

Name_____

1. Der Programmierer, der dieses Programm geschrieben hat, hat zwei Fehler mit dem eingestellten Musik-Tempo-Block gemacht. Ein Fehler ist ein Syntaxfehler, und ein Fehler ist ein logischer Fehler. Schreibe eine Erklärung für jeden dieser beiden Fehler, um sie dem Programmierer zu erklären. *Tipp*: Der Programmierer möchte, dass die Musik in einem sehr schnellen Tempo abgespielt wird.

Syntaxfehler: _____

_____

_____

Logischer Fehler: _____

_____

_____

Das ist das, was der Programmierer, der diesen Code geschrieben hat, darüber gesagt hat, was das Programm tun soll:

"Ich will, dass der Edison-Roboter sehr schnell eine Melodie spielt. Während die Musik abgespielt wird, möchte ich, dass sich der Roboter einen vollen Kreis (360 Grad) nach links dreht und dann die gleiche Strecke mit der gleichen Geschwindigkeit nach rechts zurückdreht.

Name_____

2. Das Programm funktioniert nicht so, wie der Programmierer will. Du musst das Programm debuggen und es für den Programmierer zum Laufen bringen. Teste dein Programm mit Edison und debugge solange, bis das Programm so funktioniert, wie der Programmierer es erklärt hat. Beschreibe die Fehler, die du gefunden hast und was du getan hast, um sie zu beheben.

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

Name_____

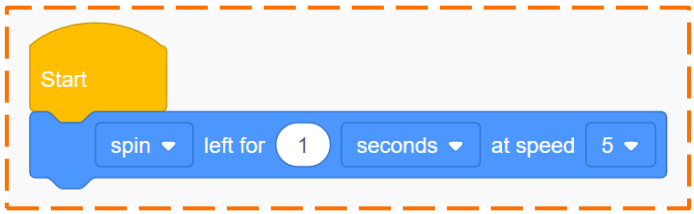
U2-2.5: Lass uns Edisons Motoren erforschen

Edison-Roboter haben zwei Motoren: einen auf der linken Seite und einen auf der rechten Seite. Die Ausgänge, die diese Motoren benutzen, sind einer der drei Haupttypen der Ausgänge deines Edison-Roboters. In EdScratch befinden sich die Blöcke, die mit den Motorausgängen zu tun haben, in der Kategorie Antrieb (Drive).

Wenn du in EdScratch ein Programm für Edison schreibst, das Blöcke aus der Drive-Kategorie verwendet, sagst du den Motoren, was sie tun sollen. Die meisten Blöcke steuern beide Motoren von Edison. Bedeutet das, dass beide Motoren das Gleiche tun?

Aufgabe 1: Den Roboter drehen

Wenn du Code schreiben wolltest, der deinem Roboter sagt, er soll sich nach links drehen, kannst du ein einfaches Ein-Block-Programm wie dieses erstellen:



Die Eingabeparameter in diesem Block teilen Edison die Richtung, die Entfernung, die Entfernungseinheiten und die Geschwindigkeit mit, die der Roboter im Programm verwenden soll.

Der Richtungs-Eingabeparameter, der ausgewählt wurde, ist spin, d.h. die gesamte Richtungseingabe ist spin links. Was ist diese Eingabe, die Edisons Motoren ausgeben soll?

In EdScratch schreibst du das Programm mit den gleichen Eingabeparametern wie auf dem Bild. Lade das Programm herunter und lasse es mit Edison auf dem Schreibtisch oder Boden laufen.

Nun starte das Programm noch einmal, aber diesmal halte Edison in deinen Händen. Fühle, wie die Räder sich bewegen. Was fällt dir auf?

1. In welche Richtung bewegt sich das linke Rad?

   _____

2. In welche Richtung bewegt sich das rechte Rad?

   _____

Edisons Motoren müssen nicht beide das Gleiche zur gleichen Zeit tun. Heißt das, du könntest ein Programm schreiben, das nur einen der Motoren bewegt? Kannst du ein Programm schreiben, das jedem Motor einzeln sagt, was er tun soll?

Name_____

Öffne die EdScratch-App und schau dir die Kategorie Drive in der Blockpalette an. Schaut euch die verschiedenen Blöcke an und schaut, ob ihr Blöcke entdecken könnt, die ihr benutzen könntet, wenn ihr nur einen Ausgang von einem von Edisons Motoren haben wolltet.

3. Was denkst du, welche Blöcke benutzen nur einen von Edisons Motoren? Warum glaubst du, dass das so ist?

_____

_____

_____

4. Du kannst Edison benutzen, um viele verschiedene Dinge zu bauen und zu erfinden. Stell dir vor, du musst etwas mit Edison bauen, das nur einen der Motoren von Edison benutzt. Was könntest du bauen? Wie würde deine Kreation den einen Motor benutzen?

Nicht vergessen

Die Räder deines Edison-Roboters können aus den Sockeln genommen werden, in denen sie sitzen. Diese Sockeln sind das, was die Motoren des Edison-Roboters tatsächlich bewegen.

_____

_____

_____

_____

_____

_____

Aufgabe 2: Richtung = vorwärts

Damit Edison so funktioniert, wie du willst, musst du sicherstellen, dass du dem Roboter alle Informationen gibst, die er braucht. Wenn der Roboter nicht alle Eingaben und Anweisungen hat, die er braucht, wird das Programm wahrscheinlich nicht so funktionieren, wie du willst. Diese Art von logischen Fehlern kann frustrierend sein, besonders wenn du denkst, du hättest dem Roboter alle notwendigen Informationen gegeben.

Eine der Hauptmöglichkeiten, wie du dem Roboter mit EdScratch Informationen gibst, sind die Eingabeparameter.

Name_____

Nicht vergessen

Jeder Eingabeparameter in einem Block gibt Edison eine andere Information, die der Roboter benötigt, um diesen Befehl ausführen zu können. Eingabeparameter sind so etwas wie die Antworten auf Fragen, die der Roboter zu dem hat, was du von ihm verlangst.

In EdScratch erhalten einige Blöcke alle Informationen, die sie benötigen, von ihren eigenen Eingabeparametern. Andere Blöcke erhalten Informationen von innerhalb ihres Blocks, benötigen aber auch Informationen von irgendwo anders im Programm.

Lass uns ein Programm für Edison erstellen, um den Roboter dazu zu bringen, seine Motoren zu bewegen. Damit dieses Programm funktioniert, gibt es vier Fragen, die du brauchst, um sicherzustellen, dass dein Programm antwortet:

Frage 1: In welche Richtung soll der Roboter gehen?

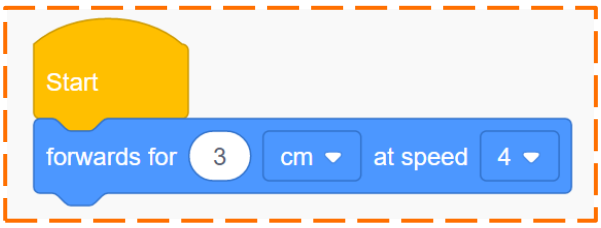
Frage 2: Wie weit soll der Roboter gehen?

Frage 3: Welche Einheiten benutzt du, um die Entfernung zu messen?

Frage 4: Wie schnell soll der Roboter fahren?

Dein Programm muss dem Roboter eine Antwort auf all diese Fragen geben.

Schau dir dieses Programm an:



Wenn du dieses Programm in einem Edison-Roboter laufen lassen würdest, hätte der Roboter dann alle Informationen, die er braucht, um zu wissen, was er tun soll? Mit

anderen Worten: Sagt dieses Programm dem Roboter die Richtung, Entfernung, Entfernungseinheiten und Geschwindigkeit, um die Motoren zu bewegen?

5. Fülle diese Tabelle aus. Wenn die Information im Programm steht, schreib den Wert dieser Eingabe in die Spalte 'Wert'. Der Wert von 'Entfernung' ist zum Beispiel die Antwort auf die Frage: 'Wie weit soll der Roboter mit diesem Programm fahren?

Information	Im Programm?	Wert
Richtung		
Entfernung		
Distanz-Einheiten		
Geschwindigkeit		

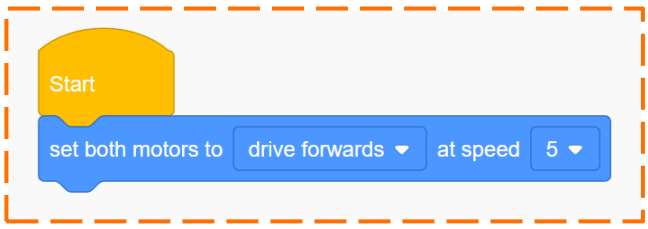
Schreibe das Programm in EdScratch, lade es herunter und lasse es in deinem Roboter laufen.

6. Was hat der Roboter getan, als du das Programm ausgeführt hast?

_____

_____

Nun schau dir das nächste Programm an:



Gibt dieses Programm dem Roboter alle Informationen, die er braucht?

Name_____

7. Fülle diese Tabelle aus. Wenn die Information im Programm ist, schreibe den Wert dieser Eingabe ein.

Information	Im Programm?	Wert
Richtung		
Entfernung		
Distanz-Einheiten		
Geschwindigkeit		

Schreibe das Programm in EdScratch, lade es herunter und lasse es in deinem Roboter laufen.

8. Was hat der Roboter getan, als du das Programm ausgeführt hast? Warum hat er sich so verhalten?

_____

_____

_____

Wie kannst du dem Roboter die restlichen Informationen geben, die er braucht, damit der Roboter seine Motoren bewegt? Experimentiere in EdScratch, um zu sehen, ob du ein Programm schreiben kannst, das den Satz beider Motoren blockiert, aber keine anderen Blöcke aus der Kategorie Antrieb verwendet und den Roboter dazu bringt, sich vorwärts zu bewegen.

Hint!

Gib nicht auf! Durch Experimentieren, Testen und Problemlösen lernen Sie neue Dinge im Programmieren.
Sich festgefahren fühlen? Denken Sie darüber nach, was Sie auf eine andere Weise versuchen möchten. Schauen Sie sich verschiedene Blöcke in EdScratch an und fragen Sie sich: „Wenn ich diesen Block verwende, wird er dann das Problem beheben, das ich zu lösen versuche?“
Wenn Sie sich nicht sicher sind, probieren Sie es aus und sehen Sie, was passiert! Achten Sie darauf, auch die Fehlerbox für Hinweise zu überprüfen!

Name_____

9. Was hast du ausprobiert? Hat es funktioniert? Schreibe auf, was du gemacht hast, was funktioniert hat, was nicht funktioniert hat und wie du dich beim Experimentieren gefühlt hast.

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

Name_____

U2-2.5a: Drehender Garten

Edisons zwei Motoren steuern jeweils eine Sockel, einen auf der rechten Seite des Roboters und einen auf der linken Seite. Wenn du willst, dass der Roboter fährt, befestigst du Räder an den Sockeln mit den Achsen der Räder. Die Motoren drehen die Achsen, wodurch sich die Räder drehen und Edison fahren kann.

Wie können die Sockel sonst noch verwendet werden?

Wenn du einen Edison-Roboter auf die Seite drehst, kannst du ihn nicht wie ein Auto fahren. Stattdessen kannst du den Roboter als angetriebene Basis für eine Erfindung verwenden!

Nicht vergessen

Die Räder Ihres Edison-Roboters können aus den Sockeln entfernt werden, in denen sie sitzen. Diese Buchsen sind das, was die Motoren des Edison-Roboters tatsächlich antreiben.

Was könnte man anstelle eines Rades in die Sockel stecken? Was passiert, wenn der Motor eingeschaltet wird?

Was ist zu tun?

In diesem Projekt musst du deine Edison-Roboter einsetzen, um einen sich drehenden Garten zu schaffen. Arbeite in einer Gruppe, um einen Garten zu entwerfen, der Edison-Roboter als Basis für Pflanzen, Blumen, Bienen, Vögel oder was auch immer du gerne in deinem sich drehenden Garten haben möchtest, benutzt.

Du kannst die Räder der Roboter abnehmen und eine andere Achse in der Sockel verwenden oder mit einem Rad als Basis bauen. Jeder Roboter muss etwas erschaffen und an ihm befestigt haben, das sich im Garten drehen kann.

Jeder Roboter muss mit EdScratch programmiert werden. Schreibe und teste deine Programme für jeden Roboter. Es kann sein, dass ihr euer Programm anpassen müsst, je nachdem, welche Art von Objekt ihr benutzt und wie ihr dieses Objekt an den Sockeln der einzelnen Roboter befestigt.

Hint!

Die Sets right motor und set left motor blocks sind sehr hilfreich, wenn sich nur ein Motor bewegen soll. Vergessen Sie nicht, dass Sie im Programm einen weiteren Block, z. B. einen Warteblock, benötigen, um die Dauer für die eingestellten Motorblöcke

Name_____

U2-2.5b: Sich drehendes Sonnensystem

Edisons zwei Motoren steuern jeweils eine Sockel, einen auf der rechten Seite des Roboters und einen auf der linken Seite. Wenn du willst, dass der Roboter fährt, befestigst du Räder an den Sockeln mit den Achsen der Räder. Die Motoren drehen die Achsen, wodurch sich die Räder drehen und Edison fahren kann.

Wie können die Sockeln sonst noch verwendet werden?

Wenn du einen Edison-Roboter auf die Seite drehst, kannst du ihn nicht wie ein Auto fahren. Stattdessen kannst du den Roboter als angetriebene Basis für eine Erfindung verwenden!

Was könnte man anstelle eines Rades in die Sockel stecken? Was passiert, wenn der Motor eingeschaltet wird?

Nicht vergessen

Die Räder Ihres Edison-Roboters können aus den Steckdosen entfernt werden, in denen sie sitzen. Diese Steckdosen sind das, was die Motoren von Edison tatsächlich bewegen.

Was ist zu tun?

In diesem Projekt müsst ihr eure Edison-Roboter einsetzen, um ein Modell des Sonnensystems zu erstellen, in dem sich die Planeten drehen können. Arbeite in einer Gruppe, um dein Modell zu bauen. Entscheidet, wie ihr die Planeten bauen wollt, ob ihr Monde einbezieht, wie groß jedes Sonnenobjekt sein wird und wie schnell sich jedes einzelne drehen wird. Wie genau könnt ihr euer Modell dem realen Sonnensystem nachbauen?

Du kannst die Räder der Roboter abnehmen und eine andere Achse in der Sockel verwenden oder ein Rad als Basis verwenden.

Jeder Roboter muss mit EdScratch programmiert werden. Schreibe und teste deine Programme für jeden Roboter. Es kann sein, dass du dein Programm anpassen musst, je nachdem, wie groß die einzelnen Objekte sind und wie du sie an den Sockeln der Roboter befestigst.

Tipp!

Der set right motor und die eingestellten set left motor sind sehr hilfreich, wenn nur ein Motor bewegt werden soll. Vergessen Sie nicht, dass Sie einen weiteren Block, wie z.B. einen wait-Block im Programm benötigen, um die Dauer für die eingestellten Motorblöcke einzustellen.

Name_____

U2-2.5c: Kartograph und Navigator

Ein Kartograph ist eine Person, die Karten erstellt. Ein Navigator ist eine Person, die herausfindet, wie man von Ort zu Ort kommt. In diesem Projekt musst du beides sein!

Was zu tun ist

Das erste, was in diesem Projekt zu tun ist, ist eine große Karte zu erstellen. Du wirst diese Karte mit deinem Edison-Roboter benutzen, also muss sie groß genug sein, damit Edison auf der Karte herumfahren kann.

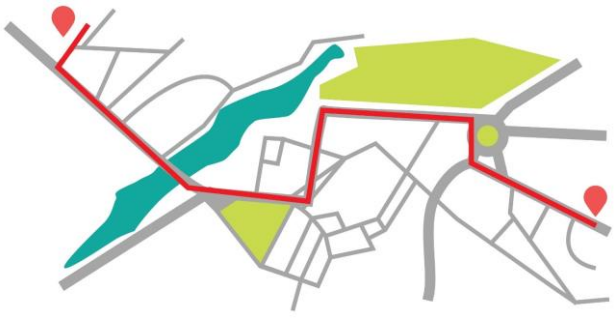
Entscheide, um welchen Ort es auf deiner Karte gehen wird. Deine Karte könnte von deiner Schule, deiner Stadt, einer fiktiven Stadt oder einem realen Ort in der Welt sein, an den du reisen möchtest. Wie auch immer du dich entscheidest, du musst deine Karte planen und dann eine Version erstellen, die groß genug ist, damit Edison-Roboter darauf fahren können.

Außerdem musst du Programmieraufgaben erstellen, die du mit deiner Karte lösen kannst. Diese Herausforderungen sollten dem Programmierer sagen, wo er den Edison-Roboter starten soll, wo der Programmierer den Roboter fertig stellen muss und welche Regeln für das Programm gelten sollen. Du könntest zum Beispiel eine Herausforderung haben, die besagt Beginne in der Schule. Bei der Eisdiele enden. Gehe nicht am Park vorbei.

Teste die Programmierherausforderungen, um sicherzustellen, dass für jede eine Lösung möglich ist. Dann tausche die Challenges mit einem Partner oder einem anderen Team. Wie viele Herausforderungen kannst du lösen?

Tipp!

In Ihren Programmregeln muss es nicht nur darum gehen, wohin man gehen und wohin man ausweichen kann. Sie können auch Regeln darüber aufstellen, wie Edison reist, z.B. rückwärts fahren oder die Geschwindigkeit, mit der sich Edison bewegt. Programmregeln, die vom Programmierer verlangen, Blöcke aus den Kategorien LEDs und Sound zu verwenden, sind ebenfalls gut!



Name_____

U2-2.5d: Autor und Regisseur

Edison kann nicht sprechen, aber das bedeutet nicht, dass du den Roboter nicht benutzen kannst, um eine Geschichte zu erzählen! In diesem Projekt musst du eine Geschichte schreiben und dann Edison 'dirigieren', damit er hilft, die Geschichte zu erzählen, als ob der Roboter ein Schauspieler in einem Theaterstück wäre.

Was zu tun ist

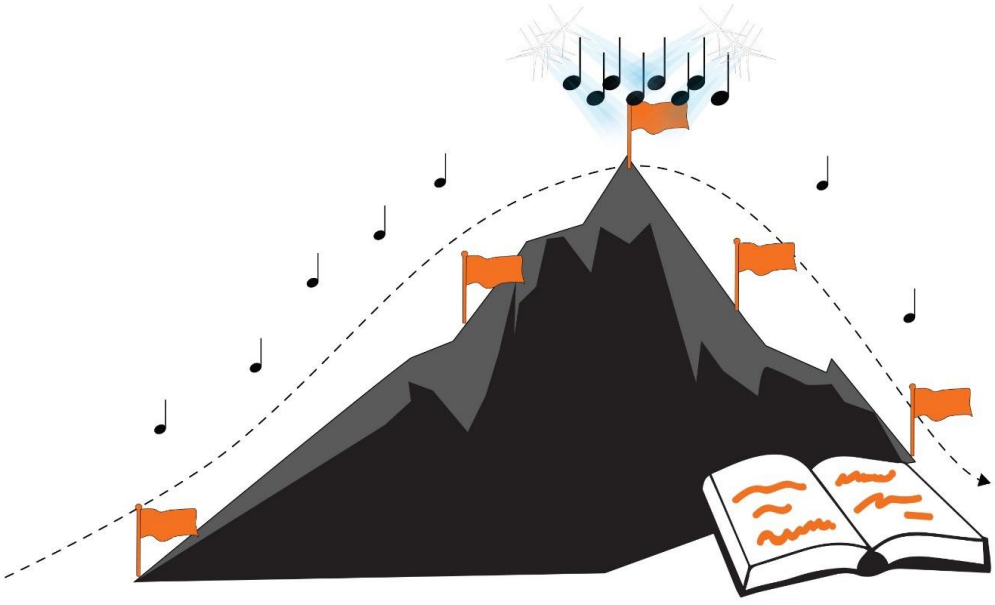
Schreibe eine Geschichte mit Hilfe einer Story Map. Edison soll dir helfen, diese Geschichte zu präsentieren. Du wirst Edison 'dirigieren', indem du den Roboter programmierst.

Tipp!

Es gibt viele Möglichkeiten, wie du dieses Projekt durchführen kannst. Du könntest eine Story Map erstellen, die groß genug ist, damit Edison weiterfahren kann und an jedem Halt auf der Story Map Aktionen ausführt. Oder du könntest Edison Aktionen im Takt mit dir ausführen lassen, während du die Geschichte laut vorliest. Oder vielleicht kannst du das sogar in einen Film verwandeln, indem du Edison bei seinen Auftritten filmst!

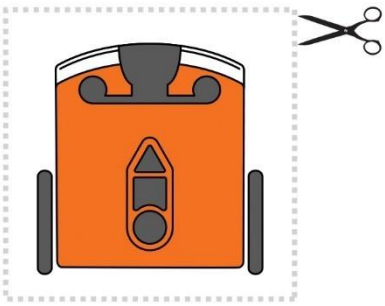
Du kannst jeden von Edisons Outputs benutzen, um die Geschichte zu erzählen. Wähle Blöcke aus den Kategorien Drive, LEDs und Sound aus. Vergiss die wait-Blöcke nicht! Du kannst Warteblöcke benutzen, um das Timing von Edison's Aktionen zu kontrollieren!

Erstelle ein Programm, dem der Roboter folgen kann, und hilf mit, die Geschichte spannend zu machen, indem du an jedem wichtigen Punkt der Geschichte etwas tust.

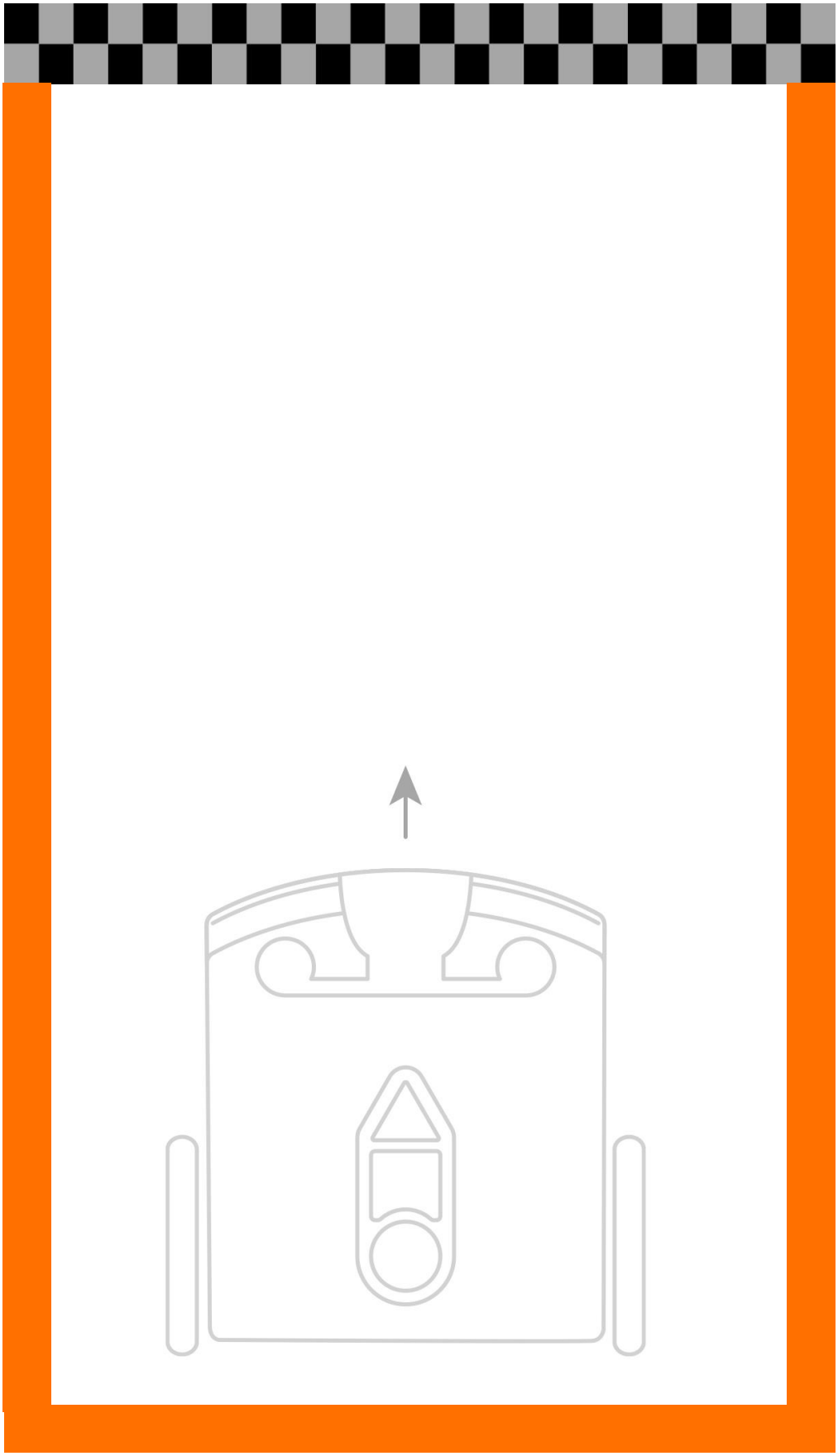


Name_____

Arbeitsblatt U2-1: Gehe Schrit



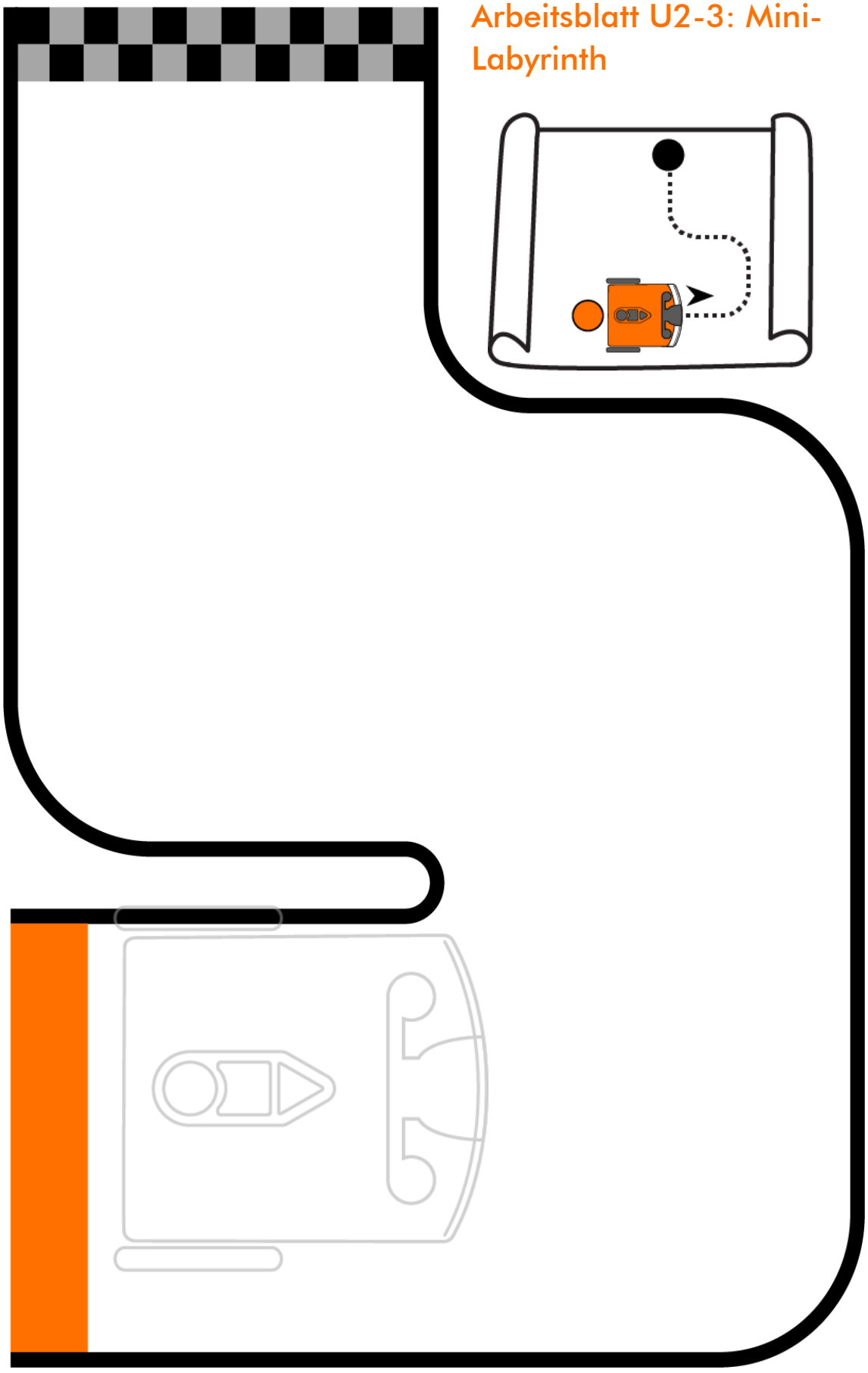
Name_____



Arbeitsblatt U2-2: Fahrstrecke

Name_____

Arbeitsblatt U2-3: Mini-Labyrinth



Name_____

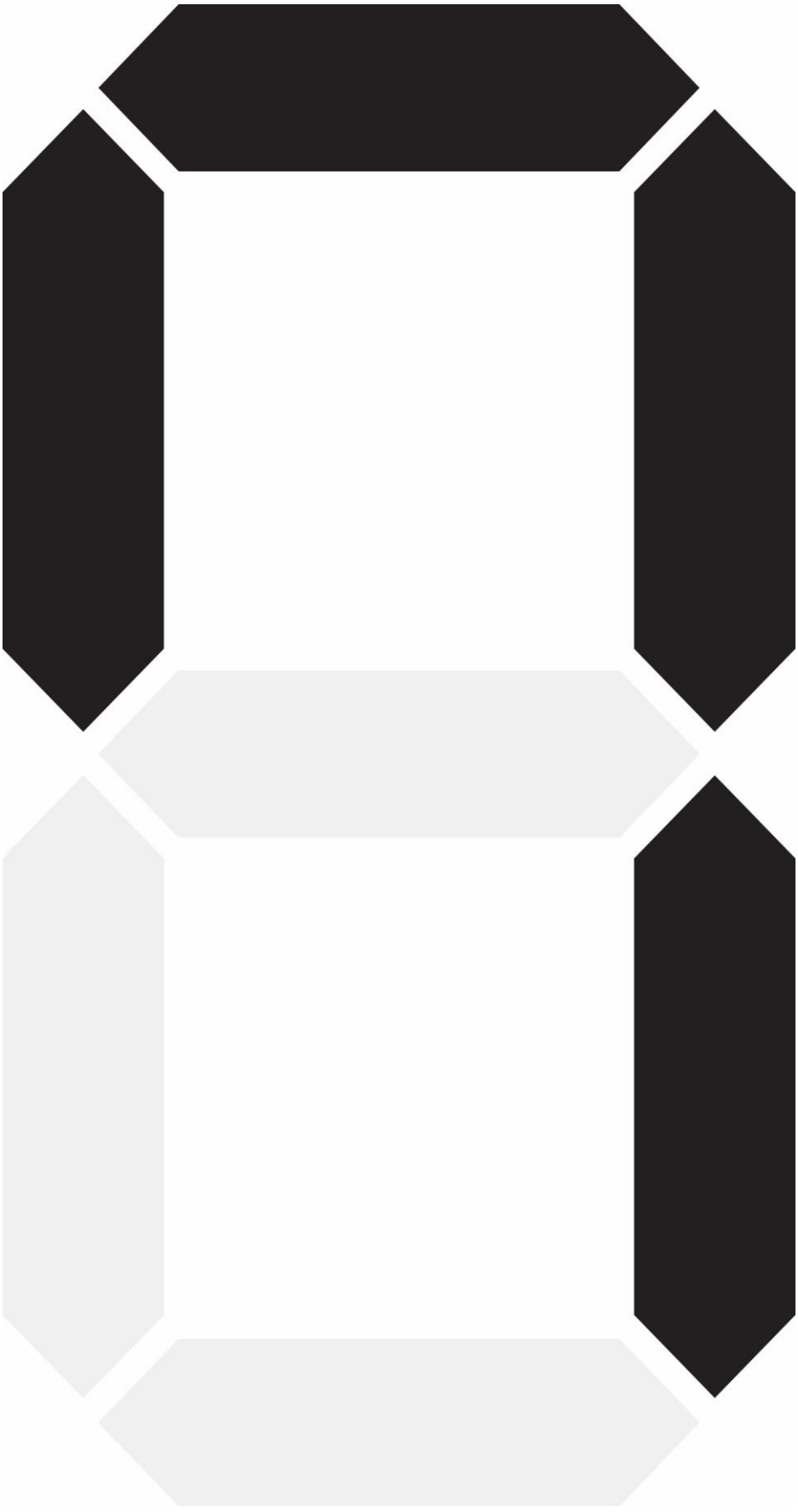
Arbeitsblatt U2-4: Digitalanzeige 2

Name_____

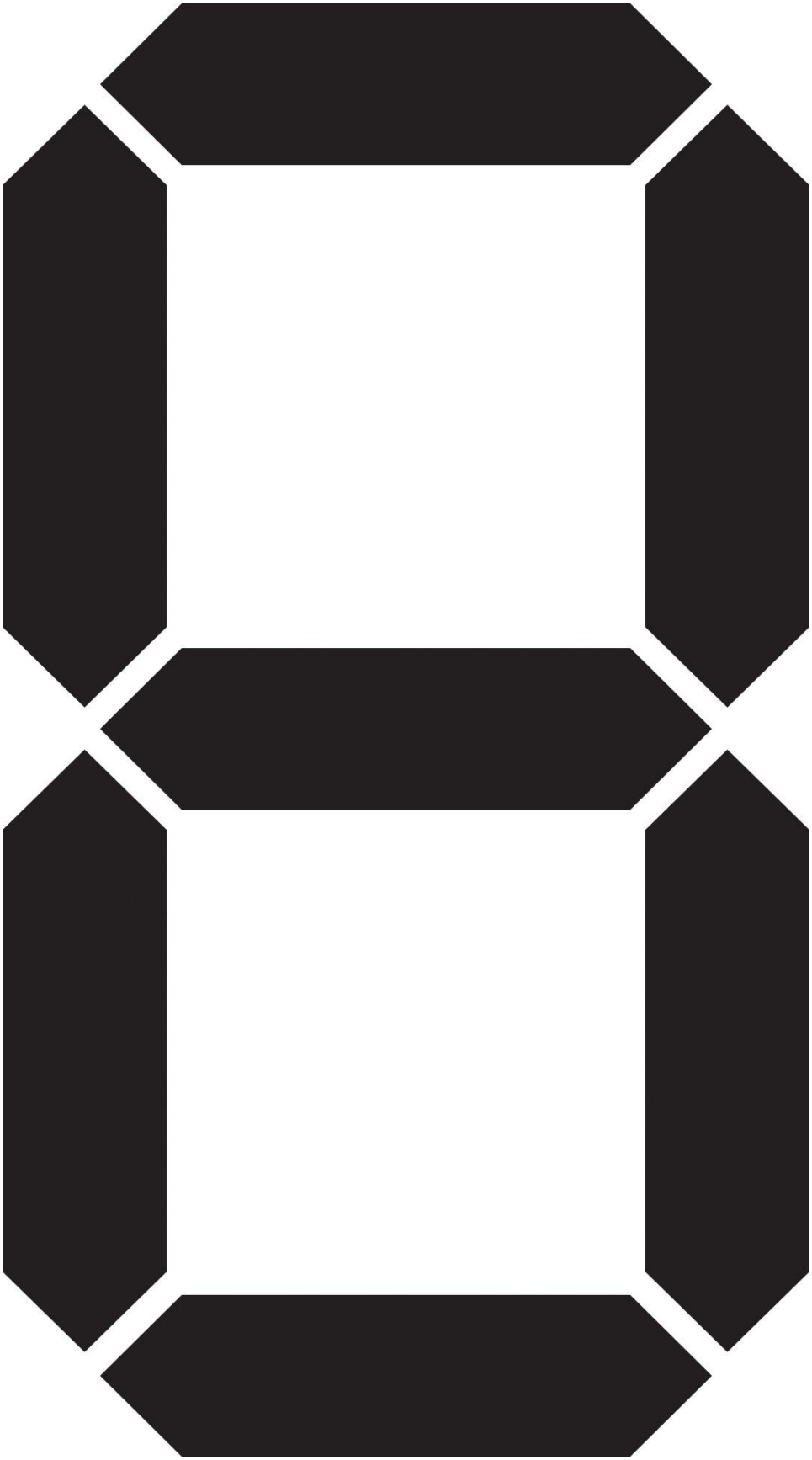
Arbeitsblatt U2-5: Digitalanzeige 5

Name_____

Arbeitsblatt U2-6: Digitalanzeige 7

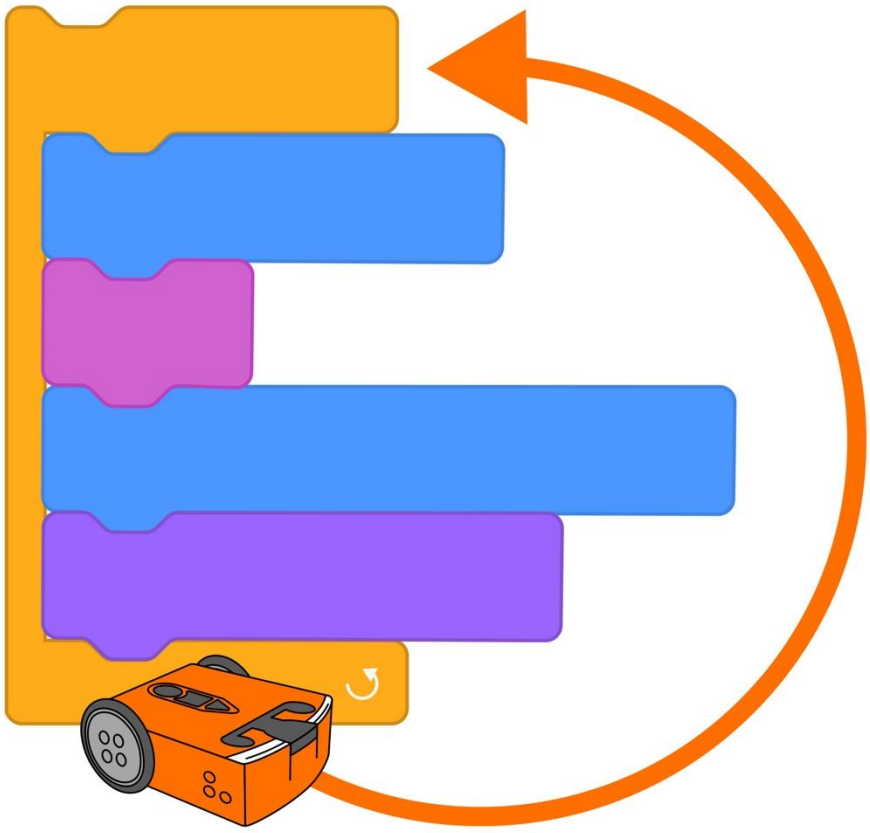


Name_____



Arbeitsblatt U2-7: Digitalanzeige 8

Einheit 3: Schleifen programmieren



Edison

Name_____

U3-1.1: sich wiederholenden Schritte

Um einen Computer, wie den Edison-Roboter, dazu zu bringen, das zu tun, was du willst, musst du sehr genaue Anweisungen geben. Du musst einen Code schreiben, der genau sagt, welche Aktionen du in welcher Reihenfolge durchführen willst.

Nicht vergessen

Wenn du in EdScratch ein Programm für deinen Edison-Roboter schreibst, sagst du dem Roboter, was er tun soll und in welcher Reihenfolge er die einzelnen Dinge tun soll. Jeder EdScratch-Block ist eine Aktion, von der du dem Roboter sagst, dass er sie ausführen soll. Die Reihenfolge, in der du die Blöcke in deinem Programm verbindest, sagt dem Roboter, in welcher Reihenfolge er jede Aktion ausführen soll. Edison führt die Aktionen einzeln nacheinander aus, beginnend mit dem obersten Block.

Aufgabe 1: In einem Quadrat fahren

Schreibe mit EdScratch ein Programm für Edison, damit dein Roboter in einem Viereck fahren kann. Dein Programm sollte nur Blöcke aus der Kategorie Drive verwenden, um die Motorausgänge zu steuern. Lade dein Programm herunter und benutze das Arbeitsblatt U3-1, um dein Programm zu testen. Stelle sicher, dass dein Programm mit Edison genau an der Stelle endet, an der es gestartet wurde.

1. Wie viele Blöcke hast du in deinem Programm, den Startblock nicht mitgezählt?

   _____

2. Schau dir die Blöcke in deinem Programm an. Was fällt dir auf? Gibt es ein Muster in den Blöcken?

_____

_____

_____

Name_____

Aufgabe 2: Benutze eine Schleife, um ein Quadrat zu fahren

Um Edison dazu zu bringen, ein Quadrat zu fahren, musst du den Roboter so programmieren, dass er jede Seite des Quadrats fährt und an jeder Ecke des Quadrats abbiegt. Du hast vielleicht bemerkt, dass dies ein Muster im Code macht: Fahre die Seite, drehe, fahre die Seite, drehe, fahre die Seite, drehe, fahre die Seite und drehe ein letztes Mal, zurück zur Ausgangsposition.

Viele Programme haben Wiederholungen, wobei ein bisschen Code immer und immer wieder verwendet wird. Dinge zu wiederholen ist eine Sache, die Computer wirklich gut können. Im Gegensatz zu einem Menschen wird es einem Computer nicht langweilig, die gleiche Sache immer und immer wieder genau gleich zu machen.

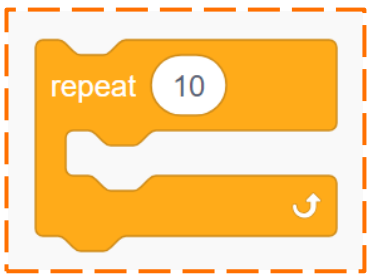
Stell dir vor, du wolltest Edison dazu bringen, dass er 100 Mal das Gleiche macht. Würdest du das Programm mit 100 sich wiederholenden Blöcken ausschreiben wollen? Würdest du es langweilig finden, das zu schreiben? Denkst du, du könntest das ganze Programm schreiben, ohne einen Fehler zu machen?

Es gibt einen einfacheren und effizienteren Weg, einen Computer dazu zu bringen, Befehle mehrmals zu wiederholen. Du kannst den Code dazu bringen, Befehle zu wiederholen, indem du etwas benutzt, das man Schleife nennt.

Fachjargon

Eine Schleife ist ein spezielles Stück Code, das einem Computer sagt, dass er etwas mehrmals wiederholen soll. Schleifen sind eine Art Kontrollstruktur, weil Schleifen andere Teile des Codes in einem Programm kontrollieren.

Beim Programmieren können wir durch die Verwendung von Schleifen andere Teile des Codes mehrfach wiederholen, ohne jeden Befehl immer wieder schreiben zu müssen. In EdScratch befinden sich die Schleifenblöcke in der Kategorie Kontrolle in der Blockpalette. Einer der Schleifenblöcke in EdScratch ist der repeat-Bloch (Wiederholungsblock):



Name_____

Es gibt verschiedene Arten von Schleifen. Der Wiederholungsblock ist eine bestimmte Schleife.

Fachjargon

Eine bestimmte Schleife ist eine Art von Schleife, die sich eine bestimmte Anzahl von Malen wiederholt. Der Wiederholungsblock in EdScratch ist ein Beispiel für eine bestimmte Schleife. Mit dem Eingabeparameter dieses Blocks teilen Sie der Schleife mit, wie oft sie wiederholt werden soll.

Wie alle Schleifenblöcke in EdScratch wickelt sich der Wiederholungsblock um andere Blöcke.

Woran liegt das?

Schau dir die Form des Wiederholungsblocks an. Siehst du, dass er eine Form hat, die ein bisschen wie ein Krokodilmaul. Andere Blöcke können in der Öffnung des ‚Mauls' dieses Blocks sitzen. Alle Blöcke, die im Wiederholungsblock sitzen, befinden sich in dieser Schleife. Alle Blöcke, die sich innerhalb der Schleife befinden, werden wiederholt.

Denke daran, Edison wird jedem EdScratch-Block einzeln folgen. Der Roboter sieht den Schleifenblock als erster und weiß, dass alle Blöcke innerhalb dieser Schleife so oft wiederholt werden müssen, wie der Eingabeparameter des Wiederholungsblocks sagt. Der Roboter wird dann die Aktion jedes Blocks innerhalb der Schleife der Reihe nach ausführen. Wenn er am unteren Ende der Blöcke in der Schleife angelangt ist, bewegt er sich wieder an den Anfang der Schleife und beginnt erneut!

Versuche einen Repeat-Block zu benutzen, um ein Programm zu erstellen, mit dem Edison ein Quadrat fahren kann. Du solltest in der Lage sein, ein Programm für Edison zu schreiben, das nur drei Blöcke nach dem Startblock verwendet, einschließlich eines Wiederholungsblocks. Lade dein Programm herunter und benutze das Arbeitsblatt U3-1, um dein Programm zu testen. Stelle sicher, dass dein Programm in Edison genau an der Stelle endet, an der es gestartet wurde.

Name_____

3. Welchen Wert musst du in dem Eingabeparameter im Wiederholungsblock haben, damit Edison ein Quadrat fährt?

_____

4. Warum musst du das als Wert haben?

_____

_____

_____

Name_____

U3-1.1a: Fahre ein Dreieck

Sogar kleine Änderungen an den Eingaben können die Ausgabe eines Programms völlig anders machen. Ein gutes Beispiel dafür ist die Änderung der Anzahl der Wiederholungen in einer Schleife. Stell dir vor, du schreibst ein Programm mit einer Schleife, die viermal wiederholt wird, und änderst dann die Eingabe so, dass sie stattdessen fünfmal wiederholt wird. Was würde passieren, wenn du das aktualisierte Programm laufen lassen würdest?

Was zu tun ist

Schreibe mit EdScratch ein Programm für Edison, damit dein Roboter in einem Dreieck fahren kann. Dein Programm muss eine bestimmte Regelkreisstruktur verwenden, also achte darauf, dass es einen Wiederholungsblock enthält. Dein Programm sollte so effizient wie möglich sein, also versuche, so wenige Blöcke wie möglich zu verwenden, während du die Aufgabe noch erledigst.

Lade dein Programm auf deinen Roboter herunter und benutze das Arbeitsblatt U3-2, um dein Programm zu testen. Stelle sicher, dass dein Programm mit Edison genau an der Stelle endet, an der es begonnen hat.

1. Wie viele Blöcke musstest du verwenden, um ein erfolgreiches Programm zu schreiben (den Startblock nicht mitgezählt)?

_____

2. Welchen Wert musst du in dem Eingabeparameter im Wiederholungsblock haben, damit Edison ein Dreieck fährt?

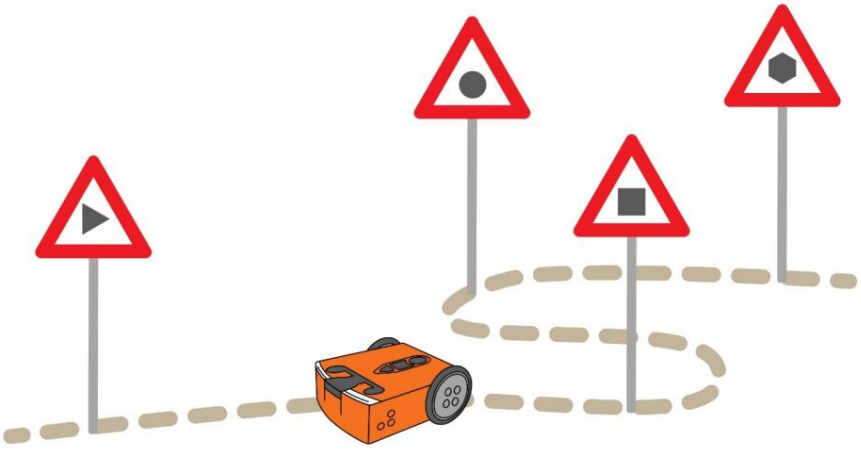
_____

3. Warum musst du das als Wert haben?

_____

_____

_____



Name_____

U3-1.1b: Fahre ein Sechseck

Sogar kleine Änderungen an den Eingaben können die Ausgabe eines Programms völlig anders machen. Ein gutes Beispiel dafür ist die Änderung der Anzahl der Wiederholungen in einer Schleife. Stell dir vor, du schreibst ein Programm mit einer Schleife, die viermal wiederholt wird, und änderst dann die Eingabe so, dass sie stattdessen dreimal wiederholt wird. Was würde passieren, wenn du das aktualisierte Programm laufen lassen würdest?

Was zu tun ist

Schreibe mit EdScratch ein Programm für Edison, damit dein Roboter in einem Hexagon fahren kann. Dein Programm muss eine bestimmte Regelkreisstruktur verwenden, also stelle sicher, dass es einen Wiederholungsblock enthält. Dein Programm sollte so effizient wie möglich sein, also versuche, so wenige Blöcke wie möglich zu verwenden, während du die Aufgabe noch erledigst.

Lade dein Programm auf deinen Roboter herunter und benutze das Arbeitsblatt U3-3, um dein Programm zu testen. Stelle sicher, dass dein Programm mit Edison genau an der Stelle endet, an der es begonnen hat.

1. Wie viele Blöcke brauchst du, um ein erfolgreiches Programm zu schreiben (den Startblock nicht mitgezählt)?

_____

2. Welchen Wert musst du im Eingabeparameter im Wiederholungsblock haben, damit Edison ein Sechseck fährt?

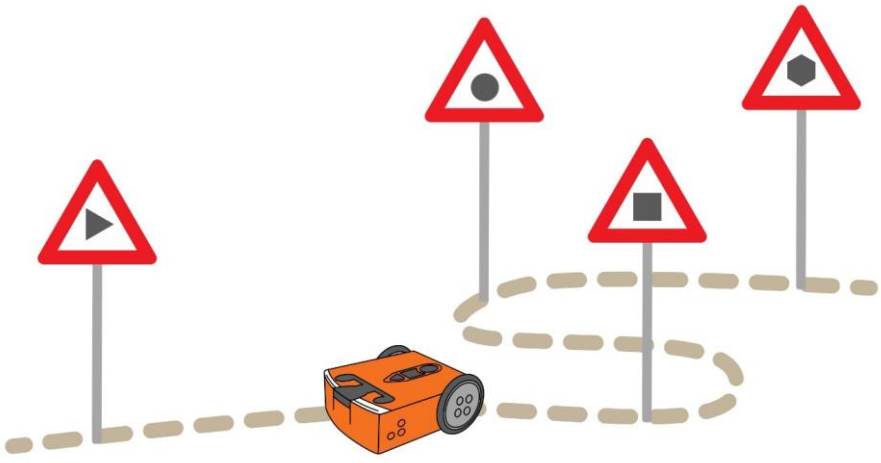
_____

3. Warum musst du das als Wert haben?

_____

_____

_____



Name_____

U3-1.1c: Wähle deine Form

Die Verwendung einer bestimmten Schleife erlaubt es dir, ein Programm zu schreiben, das Edison dazu bringt, eine Form mit nur wenigen Codeblöcken zu fahren. Du kannst kontrollieren, wie oft das Programm die Code-Befehle innerhalb der Schleife wiederholt, indem du die Eingabe des Wiederholungsblocks änderst.

Was fällt dir an der Anzahl der Seiten und Winkel auf, die eine Form hat, verglichen mit der Eingabe, die du in deiner definitiven Schleife brauchst? Kannst du dieses Muster verwenden, um ein Programm zu schreiben, das jede Form steuert?

Was zu tun ist

Wähle eine Form, die Seiten und Winkel hat, um mit deinem Edison-Roboter zu fahren.

Erstelle eine Arbeitsfläche, um dein Programm zu testen, indem du entweder deine Form auf Papier zeichnest oder sie auf dem Boden oder einem Schreibtisch mit farbigem Klebeband markierst.

Tipp!

Für diese Herausforderung solltest du vielleicht eine normale Form wählen. Eine regelmäßige Form bedeutet eine Form, bei der alle Seiten gleich sind.

Schreibe mit EdScratch ein Programm für Edison, damit dein Roboter deine Form fahren kann. Dein Programm muss eine bestimmte Schleifenkontrollstruktur verwenden, also stelle sicher, dass es einen Wiederholungsblock enthält. Dein Programm sollte so effizient wie möglich sein, also versuche, so wenige Blöcke wie möglich zu verwenden, während du die Aufgabe noch erledigst.

Lade dein Programm auf deinen Roboter herunter und teste es in deinem Arbeitsbereich.

1. Welchen Wert müsstest du im Eingabeparameter im Wiederholungsblock haben, damit Edison eine reguläre (d.h. alle Seiten sind gleich) 12-seitige Form fährt?

_____

2. Es gibt ein Muster zwischen der Anzahl der Seiten und Winkel, die eine Form hat, und der Anzahl der Male, die du eine Schleife wiederholen musst, um diese Form zu fahren. Beschreibe, wie du dieses Muster benutzt hast, um den Eingabeparameter zu bestimmen, den du im Wiederholungsblock brauchst, um Edison dazu zu bringen, deine Form zu fahren.

_____

_____

_____

_____

_____

Name_____

U3-1.1d: Einen Kreis fahren

Eine bestimmte Schleife zu benutzen, wie der Wiederholungsblock, ist hilfreich, wenn du ein Programm schreiben willst, mit dem Edison in einer Form fahren kann, weil Formen sich wiederholende Muster haben. Dir ist wahrscheinlich ein Muster zwischen der Anzahl der Seiten und Winkel einer Form aufgefallen, verglichen mit dem Input, den du in einer bestimmten Schleife in einem Programm verwenden musst, das Edison dazu bringt, diese Form zu fahren. Kann dieses Muster dir helfen, einen Kreis zu fahren, obwohl ein Kreis keine Seiten oder Winkel hat?

Was zu tun ist

Schreibe mit EdScratch ein Programm für Edison, damit dein Roboter im Kreis fahren kann. Dein Edison muss in Form eines Kreises fahren, nicht nur an einer Stelle drehen. Dein Programm muss eine bestimmte Regelkreisstruktur verwenden, also achte darauf, dass es einen Wiederholungsblock enthält. Dein Programm sollte so effizient wie möglich sein, also versuche, so wenige Blöcke wie möglich zu verwenden, während du die Aufgabe noch erledigst.

Hint!

Was fällt dir auf, wie eine Form aussieht, je mehr Seiten sie hat? Wenn du dich festgefahren fühlst, dann versuche es mit Formen, die viele Seiten haben, wie zum Beispiel ein Dekagon und ein Ikonagon. Benutze das Muster, das du siehst, um dir beim Schreiben deines Programms zu helfen.

Name_____

Lade dein Programm auf deinen Roboter herunter und benutze das Arbeitsblatt U3-4, um dein Programm zu testen.

1. Wie sieht dein Programm aus? Schreibe dein Programm unten. Achte darauf, dass du alle Eingabeparameter angibst, die du benutzt hast.

Name_____

2. Fährt dein Roboter in einem perfekten Kreis? Wenn nicht, fällt dir ein Grund ein, warum nicht?

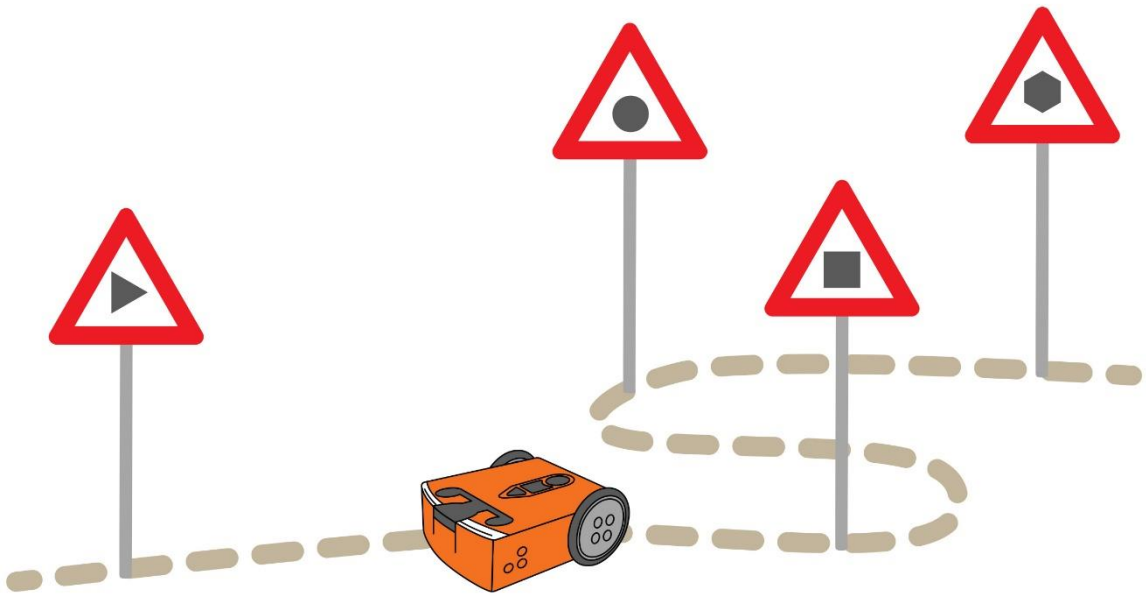
_____

_____

_____

_____

_____



Name_____

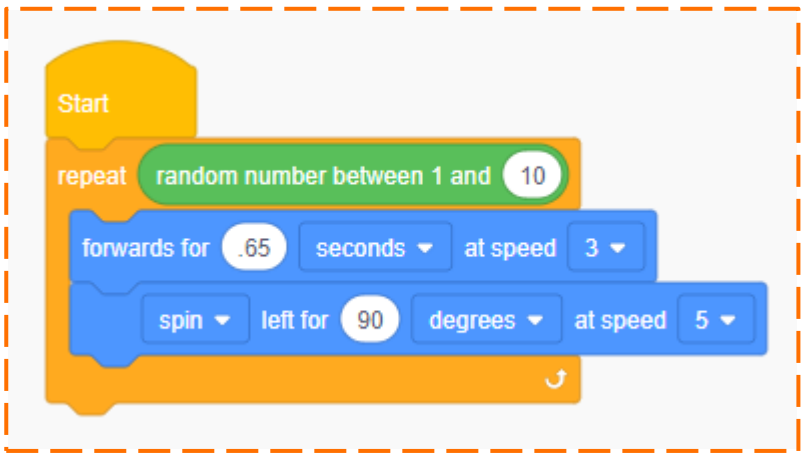
U3-1.1e: Ein Quadrat fahren

Wie viele Schleifen braucht man, um ein Quadrat zu fahren? Du weißt, dass ein Quadrat vier Seiten und vier Winkel hat, also muss der Roboter das Fahren und Wenden vier Mal wiederholen. Das heißt, wenn du ein Programm schreibst, mit dem Edison ein Quadrat mit einer bestimmten Schleife, wie dem Wiederholungsblock, befahren kann, musst du die Schleife viermal wiederholen lassen.

Was passiert, wenn sich die Schleife nur dreimal wiederholt? Was ist, wenn sie sich neun Mal wiederholt?

Was ist zu tun?

Schau dir dieses EdScratch Programm an:



Dieses Programm benutzt einen speziellen Eingabeparameter für den Wiederholungsblock: den Zufallszahlenblock! Dieser Block weist Edison an, eine Zahl zwischen 1 und 10 nach dem Zufallsprinzip zu wählen. So oft wird der Roboter den Code innerhalb des Wiederholungsblocks wiederholen.

Schreibe das Programm in EdScratch. Lade dein Programm auf deinen Roboter herunter und benutze das Arbeitsblatt U3-1, um dein Programm zu testen. Versuche das Programm mehrmals auszuführen, um zu sehen, was passiert.

1. Was ist passiert, als du das Programm ausgeführt hast? Ist jedes Mal dasselbe passiert? Warum oder warum nicht?

_____

_____

_____

_____

_____

Name_____

U3-1.1f: Doodle-bot-Herausforderung

Du kannst Edison so programmieren, dass er in Mustern fährt, die viele verschiedene Formen ergeben. Wenn du einen Stift an deinem Roboter anbringst, kannst du Edison dazu bringen, auch diese Formen zu zeichnen!

Was ist zu tun?

Befestige einen Stift an einem Edison-Roboter. Du kannst EdCreate Teile oder jedes andere Material benutzen, das dir gefällt.

Schreibe ein Programm in EdScratch, so dass dein Roboter, wenn du das Programm ausführst, eine Form zeichnet. Wenn du deinen Stift gebaut und deinen Roboter programmiert hast, versuche das Programm auszuführen, um deine Form zu sehen!

1. Schreibe über das, was du bei dieser Herausforderung erlebt hast. Auf welche Probleme bist du gestoßen? Was hast du getan, um diese Probleme zu überwinden? Wie hast du den Stift angebracht? Welche Form hast du versucht zu zeichnen? Hat es funktioniert?

_____

_____

_____

_____

_____

_____

_____

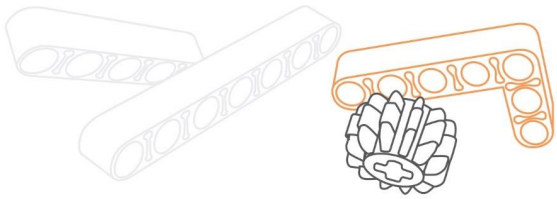
_____

_____

_____

_____

_____



Name_____

U3-1.2: Schleifen und Sequenzen

Schleifen sind eine sehr nützliche Kontrollstruktur im Coding. Schleifen können dabei helfen, Programme effizienter zu machen, indem sie es dir ermöglichen, Befehle zu wiederholen, ohne die gleichen Codeblöcke mehrmals schreiben zu müssen.

Wenn du Schleifen in Programmen verwendest, musst du trotzdem sorgfältig über die Reihenfolge nachdenken. Das gilt besonders, wenn du Programme machst, die etwas Code innerhalb einer Schleife und etwas Code außerhalb der Schleife haben.

Nicht vergessen

Sequenz bedeutet, der Reihe nach zu gehen, Schritt für Schritt.

Lass uns versuchen, ein Programm zu machen, das Edison ein Viereck fahren lässt. Dein Programm wird einen Teil des Codes innerhalb einer Schleife haben müssen, aber ein Teil des Codes muss außerhalb der Schleife liegen.

Probiert es aus!

Ein Quadrilateral ist eine vierseitige Form. Quadrate sind Vierecke, aber nicht alle Vierecke sind Quadrate. Schau dir das Viereck auf dem Arbeitsblatt U3-5 an. Dieses Viereck hat vier Seiten und vier Winkel, aber sie sind nicht alle gleich.

Du musst ein Programm für Edison mit EdScratch schreiben, damit dein Roboter die Form des Vierecks auf dem Arbeitsblatt U3-5 fahren kann. Dein Programm sollte Blöcke aus der Kategorie Drive verwenden, um die benötigten Motorleistungen zu erzeugen. Dein Programm muss auch eine Schleife aus der Kategorie Control verwenden.

Du musst den besten Startplatz für Edison auf dem Arbeitsblatt herausfinden. Stelle sicher, dass dein Programm Edison auch genau an der Stelle endet, an der es begonnen hat.

Tipp!

Denke über die Reihenfolge der Aktionen nach, die Edison ausführen soll. Denke daran, wenn du ein EdScratch-Programm für Edison erstellst, startet der Roboter mit dem obersten Block und führt jede Aktion der Reihe nach aus, eine nach der anderen.

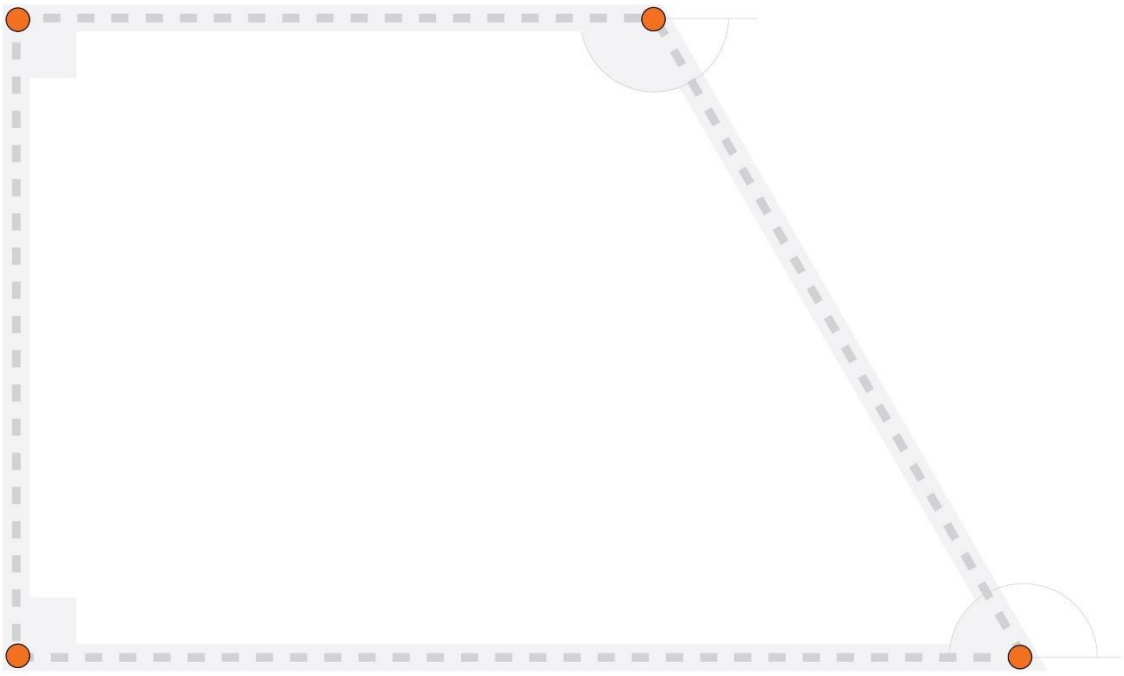
Schreibe dein Programm in EdScratch. Dann lade dein Programm herunter und benutze das Arbeitsblatt U3-5, um es zu testen.

Name_____

Tipp!

Du willst dein Programm immer noch so effizient wie möglich machen, also versuche, so wenige Codeblöcke wie möglich zu verwenden.

1. Wo auf dem Viereck hast du mit Edison angefangen? Markiere, wo du mit Edison angefangen hast, einschließlich der Richtung, in die du den Roboter fahren ließest.



2. Schau dir deinen Startpunkt und dein Programm an. Wie hat sich die Verwendung dieses Startpunktes auf den Ablauf deines Programms ausgewirkt?

_____

_____

_____

_____

Name_____

U3-1.3: Unendliche Schleifen

Schleifen lassen uns Schritte in einem Programm wiederholen, ohne immer wieder den gleichen Code schreiben zu müssen. Wenn du möchtest, dass ein Programm dasselbe viele Male macht, ist es einfacher, eine Schleife zu benutzen, als jeden Befehl immer wieder schreiben zu müssen. Das macht unseren Code effizienter, weil du dem Computer sagen kannst, dass er das Gleiche mit viel weniger Code machen soll.

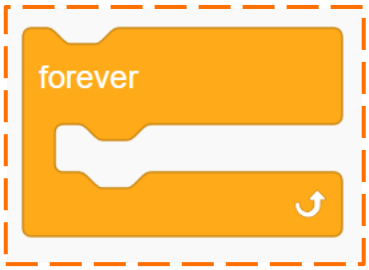
Bei vielen Programmen mit sich wiederholenden Befehlen weißt du, wie oft das Programm in einer Schleife wiederholt werden soll. Wenn du zum Beispiel deinen Roboter dazu bringen willst, in einem Viereck zu fahren, weißt du, dass du Edison fahren und vier Mal wenden musst. In diesem Programm kannst du eine bestimmte Schleife verwenden, die sich viermal wiederholt.

Nicht vergessen

Es gibt verschiedene Arten von Schleifen. Eine bestimmte Schleife ist eine Art von Schleife, die sich für eine bestimmte Anzahl von Malen wiederholt. Der Wiederholungsblock in EdScratch ist ein Beispiel für eine bestimmte Schleife.

Um eine bestimmte Schleife zu verwenden, musst du wissen, wie oft die Schleife wiederholt werden muss. Und wenn du das nicht weißt? Oder, was ist, wenn du ein Programm machen willst, das ewige Schleifen macht?

Um etwas in EdScratch ewig wiederholen zu lassen, musst du einen speziellen Schleifenblock in der Kategorie Control in der Blockpalette verwenden, der forever-Block genannt wird:



Der forever-Block ist eine unbestimmte Schleife.

Name_____

Fachjargon

Eine unbestimmte Schleife ist eine Art von Schleife, die sich für eine unbestimmte Anzahl von Malen wiederholt. Der Forever-Block in EdScratch ist ein Beispiel für eine unbestimmte Schleife. Dieser Schleifenblock weist Edison an, die Codeblöcke innerhalb dieser Schleife für immer zu wiederholen.

Du kannst dir vorstellen, dass der forever-Block in EdScratch genauso funktioniert wie der Wiederholungsblock, aber mit dem Eingabeparameter für die Anzahl der Schleifen auf unendlich gesetzt!

Die Form eines Blocks in EdScratch kann dir einige Hinweise darauf geben, wie du ihn in der Sprache verwendest. Schau dir die Form des forever-Blocks an. Genau wie all die anderen Schleifenblöcke in EdScratch wickelt sich der Forever-Block um andere Blöcke. Alle Blöcke, die sich innerhalb des Schleifenblocks befinden, werden wiederholt. Was fällt dir noch an der Form dieses Blocks auf?

1. Wenn du ein Programm mit einem forever-Block schreibst, glaubst du, dass du nach der Schleife Befehle für Edison hinzufügen kannst? Warum oder warum nicht?

_____

_____

_____

_____

Name_____

Probier es aus!

Lass uns Edison in eine Eieruhr verwandeln! Benutze den Forever-Block, um ein Programm zu schreiben, so dass Edison eine bestimmte Anzahl von Sekunden wartet und dann ununterbrochen Alarm schlägt.

Tipp!

Du kannst jederzeit ein Programm stoppen, indem du den Stop-Knopf (quadratisch) deines Edison-Roboters drückst.

Denke über die Reihenfolge der Dinge nach, die passieren müssen, damit das Eieruhrprogramm funktioniert. Was muss in der Schleife stehen? Was muss außerhalb der Schleife sein? Lade dein Programm herunter und teste es mit deinem Roboter.

2. Wie sieht dein Programm aus? Schreibe dein Programm unten. Achte darauf, dass du die von dir verwendeten Eingabeparameter angibst.

Name_____

U3-1.3a: Ohrwurm

Ein Ohrwurm ist ein Lied, das sich für immer in deinem Kopf festsetzt. Bei dieser Aktivität musst du Edison einen Ohrwurm geben, indem du den Roboter mit einer Melodie und einer ewigen Blockade programmierst!

Was zu tun ist

Programmiere Edison so, dass er ein Lied oder eine Melodie immer und immer wieder spielt, indem er einen Forever-Loop-Block benutzt. Schreibe dein Programm in EdScratch, lade es dann herunter und teste es mit deinem Roboter.

Tipp!

Du kannst jederzeit ein Programm stoppen, indem du den Stop-Knopf (quadratisch) deines Edison-Roboters drückst.

Name_____

U3-1.4: Das Stapeln und Verschachteln von Schleifen

Wenn du Programme mit Schleifen schreibst, kannst du effizienter arbeiten, weil du einen Computer dazu bringen kannst, Aktionen zu wiederholen, ohne die Befehle mehrfach schreiben zu müssen. Mit Schleifen kannst du einem Programm auch sagen, dass es für immer etwas tun soll, was du nicht tun könntest, wenn du jeden Befehl einzeln aufschreiben müsstest.

Du kannst auch mehr als eine Schleife in einem Programm verwenden, und du kannst die Schleifen auf verschiedene Weise nutzen: indem du die Schleifen aufeinander stapelst (stacking) oder Schleifen in andere Schleifen verschachtelst (nesting).

Fachjargon

In blockbasierten Programmiersprachen wie EdScratch wird das Zusammenfügen von Blöcken manchmal als Stapeln (stacking) von Blöcken und ein Programm manchmal als Stack oder Blockstapel bezeichnet. Deshalb kann man sagen, dass man die Schleifen stapelt, wenn man mehrere Schleifen nacheinander in einem Programm benutzt.

Du kannst auch einen Schleifenblock in einen anderen Schleifenblock legen. Dies nennt man verschachtelte Schleifen (nesting loops).

Warum solltest du Schleifen in Stapeln verwenden oder sie ineinander verschachteln? Wenn du auf diese Weise mehrere Schleifen zusammen verwendest, kannst du Programme mit sich wiederholenden Mustern schreiben. Du kannst sogar Programme mit Mustern schreiben, die sich in anderen Mustern wiederholen.

Name_____

Woran liegt das?

Denke an einen Wecker auf einem Handy. Der Wecker kann so eingestellt werden, dass er morgens um 7:00 Uhr morgens losgeht. Du kannst das Handy so einstellen, dass der Wecker jeden Tag wiederholt wird. Wenn der Wecker losgeht, piepst er eine bestimmte Anzahl von Malen an und aus. Wenn du den Wecker abschaltest, stoppt er für eine bestimmte Zeit, dann schaltet er sich wieder ein und piepst eine bestimmte Anzahl von Malen an und wieder aus.

Kannst du sehen, dass es sich wiederholende Muster in anderen sich wiederholenden Mustern gibt?

Dies ist ein Beispiel, bei dem es sehr hilfreich wäre, gestapelte und verschachtelte Schleifen zu verwenden, um ein Programm zu schreiben. Denn mit gestapelten und verschachtelten Schleifen kannst du ganze Sätze von Befehlen innerhalb deines Programms wiederholen.

Das Stapeln und Verschachteln von Schleifen hat unterschiedliche Verwendungszwecke. Durch das Stapeln von Schleifen können wir Programme schreiben, die Edison dazu bringen, verschiedene Sätze von Aktionen mehrmals auszuführen und dann zu einem neuen Satz von sich wiederholenden Aktionen überzugehen. Durch das Ineinanderschachteln von Schleifen können wir jedoch Programme schreiben, die Edison dazu bringen, ganze Muster mehrfach zu wiederholen.

Name_____

Aufgabe 1: Was wird passieren?

Programme, die mehrere Schleifen verwenden, besonders verschachtelte Schleifen, mögen auf den ersten Blick etwas verwirrend erscheinen. Um zu verstehen, was das Programm tun wird, musst du über jede Aktion nachdenken, die nacheinander stattfinden wird.

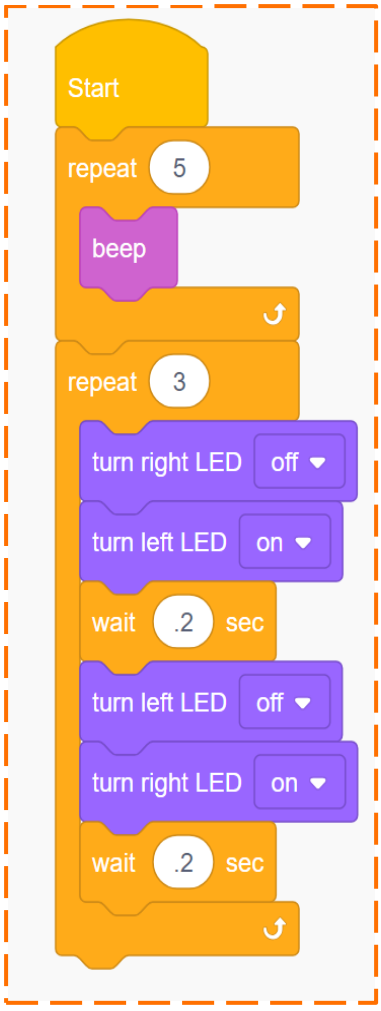
Nicht vergessen

Wenn du in EdScratch ein Programm für Edison erstellst, startet der Roboter mit dem obersten Block und führt jede Aktion nacheinander aus. Sobald er einen Block abgeschlossen hat, geht er zum nächsten Block über. Dies gilt für alle EdScratch-Programme - egal ob es null Schleifen, eine Schleife oder mehrere Schleifen gibt!

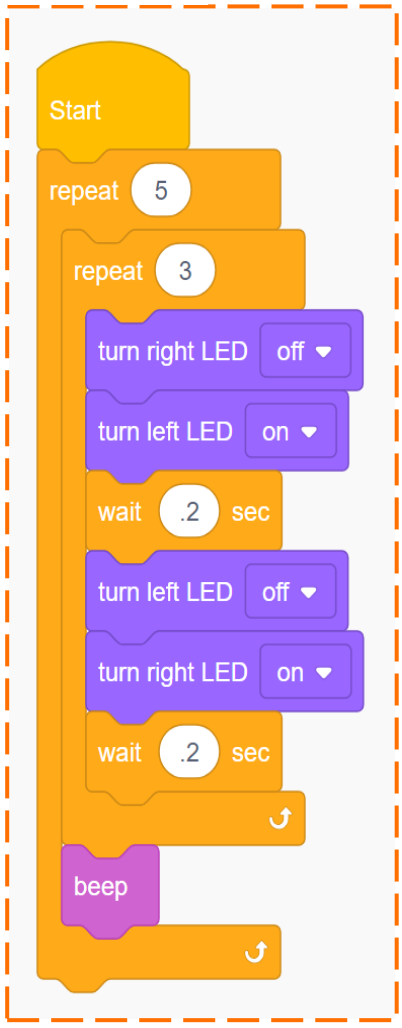
Schau dir die folgenden Programme an und beantworte die Fragen darüber, was in jedem Programm passieren wird.

Name_____

Program 1:



Program 2:



1. In Programm 1, wie oft wird die rechte LED aufleuchten?

   _____

2. In Programm 1, das als erstes beendet wird: alle Pieptöne oder alle LED blinken?

   _____

3. In Programm 2, wie oft wird die rechte LED aufleuchten?

   _____

4. In Programm 2, das als erstes beendet wird: alle Pieptöne oder alle LED blinken?

   _____

Name_____

Tipp!

Kannst du mitverfolgen, was in jedem Programm passiert? Wenn du deine Antworten doppelt überprüfen möchtest, versuche, jedes Programm in EdScratch zu schreiben und es dann auf deinen Edison-Roboter herunterzuladen. Starte das Programm, um zu sehen, was passiert.

Aufgabe 2: Fahre das Muster

So wie eine Schleife es dir ermöglicht, ein Muster mehrmals zu wiederholen, so kannst du mit verschachtelten Schleifen mehrere Muster wiederholen! Für diese Aktivität musst du dein Arbeitsblatt U3-6 verwenden.

5. Schau dir das Muster auf dem Arbeitsblatt U3-6 an. Wie würdest du das Muster beschreiben?

_____

_____

_____

Du musst ein Programm schreiben, damit Edison das Muster auf dem Arbeitsblatt U3-6 fährt. Dein Programm kann Edison mehr als einmal über dieselbe Linie fahren lassen, aber der Roboter muss alle Linien berühren.

6. Wie denkst du, dass du eine verschachtelte Schleife verwenden kannst, um ein effizientes Programm für deinen Edison-Roboter zu schreiben, damit er das Muster auf dem Arbeitsblatt fährt?

_____

_____

_____

Name_____

Versuche, ein EdScratch-Programm zu schreiben, damit dein Edison-Roboter das Muster auf dem Arbeitsblatt U3-6 fahren kann. Teste deine Idee, eine verschachtelte Schleife zu benutzen, um zu sehen, ob es funktioniert.

Tipp!

Du kannst ein Programm schreiben, das den Aktivitätenbogen mit nur fünf EdScratch-Blöcken ausfüllt!

Name_____

U3-1.4a: Edison der Designer

Viele Dinge, die mit Computerprogrammen laufen, haben sich wiederholende Muster. Es gibt auch viele Programme, die Muster haben, die sich innerhalb anderer Muster wiederholen. Diese Programme verwenden oft verschachtelte Schleifen, um ganze Sätze von Befehlen innerhalb eines Programms zu wiederholen.

Was zu tun ist

Versuche, mit Schleifen ein Programm für deinen Roboter zu schreiben, das Edison dazu bringt, ein Muster zu fahren. Wenn dieses Design ein Muster mit einem sich wiederholenden Muster darin hat, versuche es mit einer verschachtelten Schleife.

Schau dir das Arbeitsblatt U3-7 an und wähle eines der Muster aus. Für diese Aktivität musst du einen Arbeitsbereich erstellen, um dein Programm zu testen. Erstelle eine Arbeitsfläche, die groß genug ist, um dein Programm mit Edison zu testen. Du kannst das Muster auf ein großes Blatt Papier zeichnen oder es mit dunklem Klebeband auf dem Boden markieren. Kopiere das Muster auf deine Arbeitsfläche. Dann schreibst du ein Programm in EdScratch, das Edison dazu bringt, dieses Muster zu fahren.

Tipp!

Festgefahren? Versuche, das Muster in kleinere Abschnitte zu zerlegen und Code zu schreiben, um Edison dazu zu bringen, jeden Teil des Musters zu fahren. Verbinde alle Teile miteinander, damit Edison das ganze Muster steuert. Das kann dir helfen, Stellen zu finden, an denen sich der Code wiederholt. Mach dein Programm effizienter, indem du sich wiederholenden Code durch Schleifen ersetzt.

Wenn es ein Muster innerhalb eines Musters gibt, versuche es unbedingt mit einer verschachtelten Schleife!

Mini-Herausforderung!

Wenn du möchtest, kannst du auch ein eigenes Muster entwerfen, das du für diese Aktivität verwenden kannst. Stelle sicher, dass dein Entwurf ein Muster hat, das sich innerhalb eines anderen Musters wiederholt. Teste deinen Entwurf, indem du ein Programm schreibst, das Edison dazu bringt, dein Muster zu fahren.

Braucht dein Muster verschachtelte Schleifen?

Name_____

U3-1.4b: Tanzparty!

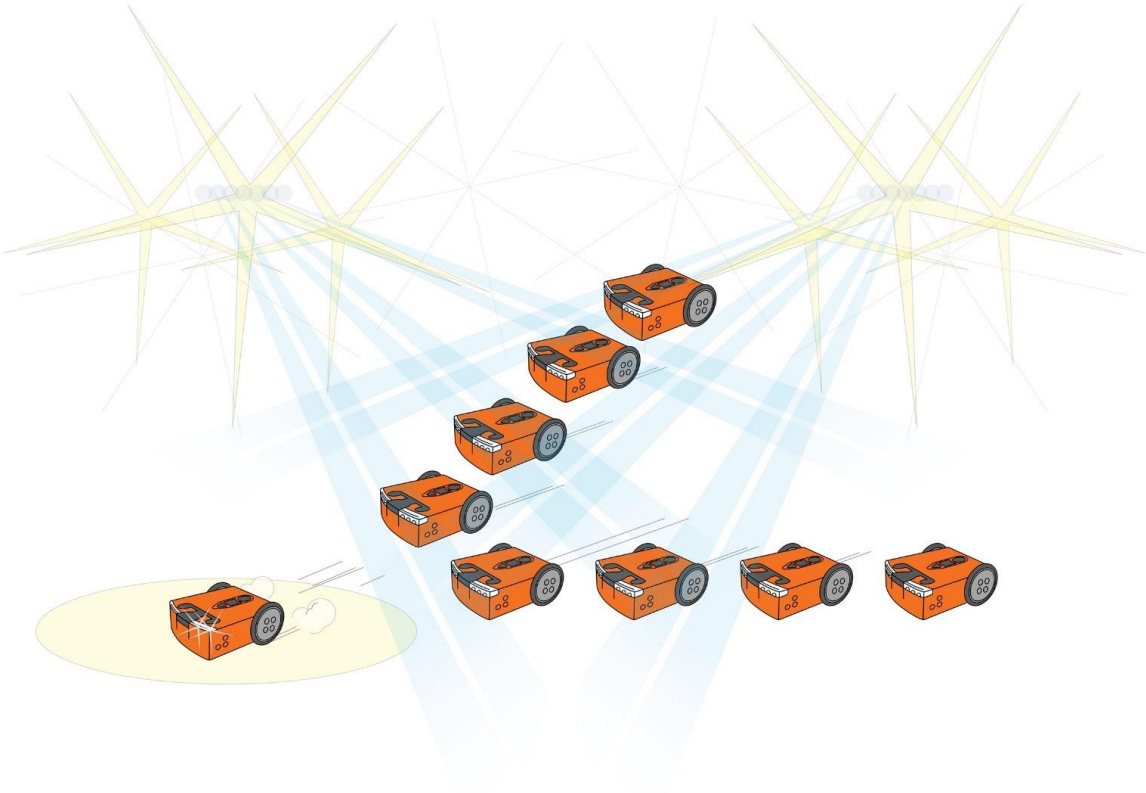
Musik und Tanz haben oft sich wiederholende Muster. Der Refrain eines Liedes wiederholt sich und die Noten innerhalb eines Liedes haben oft auch ein Muster. Viele Tänze wiederholen auch Bewegungen. All diese Wiederholungen bedeuten, dass dein Programm, wenn du eine Roboter-Tanzparty veranstaltest, wahrscheinlich Loops braucht!

Was zu tun ist

Arbeitet zusammen, um mehrere Edison-Roboter für eine Tanzparty zu programmieren. Du kannst entweder alle Edison-Roboter zu einem Lied tanzen lassen, das du von einem anderen Gerät aus spielst, oder einen Edison das Lied spielen lassen, während die anderen Roboter tanzen.

Benutze irgendeinen der Edison-Ausgänge (mit Blöcken aus den Kategorien Drive, LEDs und Sound) zusammen mit Loop-Blöcken aus der Kategorie Control, um ein Tanzprogramm zu erstellen. Werden alle Roboter das Gleiche zur gleichen Zeit tun? Ist ein Roboter der Star, während die anderen Roboter als Backgroundtänzer fungieren?

Das hängt von dir und deinem Team ab!



Name_____

U3-2.1: Das Unterbrechen des Hauptprogramms

Verschiedene Computerprogrammiersprachen haben unterschiedliche Syntaxen oder Regeln, wodurch sie ein bisschen anders aussehen und sich anders anfühlen. Unabhängig von der Syntax arbeiten jedoch alle Computersprachen mit der gleichen zugrunde liegenden Logik. Deshalb verhalten sich alle Computerprogramme auf ähnliche Weise und folgen der Kernlogik der Programmierung, wie Sequenzen.

Nicht Vergessen!

Logik ist die organisierte Art, Dinge zu tun, die für einen Computer Sinn macht. Die Logik bestimmt den Ablauf eines Programms.

Syntax sind die Regeln, nach denen eine Programmiersprache funktioniert.

In EdScratch besteht der logische Ablauf eines Programms darin, mit dem obersten Block zu beginnen und jede Aktion Block für Block abzuschließen. Programme mit Schleifen folgen ebenfalls der Sequenz. Wenn das Programm einen Schleifenblock sieht, führt es die Befehle innerhalb dieser Schleife der Reihe nach aus. Wenn es am unteren Ende der Schleife angelangt ist, geht es zum oberen Ende der Schleife zurück und beginnt erneut. Auch wenn Schleifenblöcke Programme etwas anders aussehen lassen, folgen diese Programme immer noch dem logischen Ablauf von oben nach unten.

Es gibt eine Möglichkeit, diesen sequentiellen Fluss zu unterbrechen. Du kannst den Ablauf eines Computerprogramms unterbrechen, indem du einen Interrupt verwendest.

Fachjargon

Ein Interrupt ist ein spezieller Teil des Codes, der den Fluss des Hauptcodes stoppt. Es wird Interrupt genannt, weil es den Hauptcode unterbricht. Ein Interrupt wird normalerweise verwendet, um den Hauptcode zu pausieren, um eine Subroutine auszuführen.

Eine Subroutine ist ein bestimmter Satz Code, der vom Hauptprogramm getrennt ist. Man kann sich eine Subroutine als ein Miniprogramm vorstellen.

Um zu verstehen, wie Unterbrechungen funktionieren, müssen wir verstehen, was unterbrochen wird.

Name_____

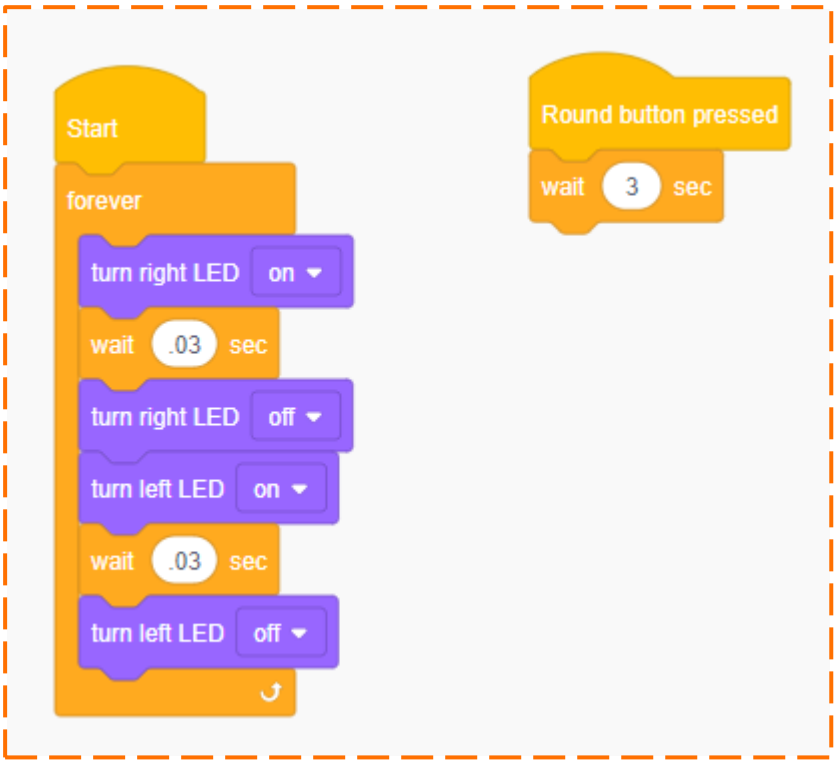
Was ist das Hauptprogramm?

In EdScratch ist das Hauptprogramm das, was mit dem gelben Startblock (Start) verbunden ist.

Immer wenn du ein Programm für Edison schreibst, musst du mindestens einen Block im Hauptprogramm an den Startblock angehängt haben. Wenn du ein Programm mit deinem Edison-Roboter ausführst, indem du auf die Play-Taste (Dreieck) drückst, läuft dieses Hauptprogramm Block für Block in sequentieller Reihenfolge, bis es das Ende des Programms erreicht.

Eine Unterbrechung kann diesen Ablauf unterbrechen.

Schau dir das folgende Programm an:



Name_____

Dieses Programm hat zwei Teile: das Hauptprogramm und die Subroutine.

1. Was sagt das Hauptprogramm Edison, was er tun soll? Hinweis: Nur die Blöcke, die an den Startblock angehängt sind, sind im Hauptprogramm.

_____

_____

_____

Neben dem Hauptprogramm gibt es auch eine Unterroutine. Was bewirkt, dass das Hauptprogramm unterbrochen wird und die Subroutine ausgeführt wird?

Subroutinen laufen nur, wenn ein bestimmtes Ereignis eintritt.

Fachjargon

Beim Programmieren ist ein Ereignis etwas, das außerhalb des Programmcodes geschieht und die Art und Weise beeinflusst, wie das Programm läuft. Ein Ereignis kann z.B. das Drücken einer Taste oder die Übertragung von Informationen von einem Sensor sein.

In EdScratch musst du einen Block aus der Kategorie Ereignisse (Event) am Anfang jeder Subroutine verwenden.

2. Schau dir die Blöcke in der Kategorie Ereignisse in EdScratch an. Was fällt dir an der Form dieser Blöcke auf?

_____

_____

Die Ereignisblöcke sind Unterbrechungen, die dem Programm sagen, dass es nach diesem bestimmten Ereignis Ausschau halten soll. Wenn das Ereignis eintritt, unterbricht der Ereignisblock das Hauptprogramm sofort und führt die Unterroutine aus. Sobald der Code der Subroutine beendet ist, kehrt das Programm an die Stelle zurück, an der es im Hauptprogramm aufgehört hat.

Name_____

Woran liegt das?

Wir benutzen Interrupts in der Programmierung, weil Interrupts es einem Programm erlauben, jederzeit auf ein Ereignis zu reagieren, während das Programm läuft. Durch die Verwendung von Interrupts musst du nicht vorhersagen, wann ein Ereignis eintreten wird. Ohne Interrupts müsstest du genau wissen, wann etwas passieren wird, auch wenn dieses Ereignis außerhalb deiner Kontrolle liegt!

Probier es aus!

Schreibe ein Programm in EdScratch, das sowohl das Hauptprogramm als auch die Subroutine so enthält, wie sie auf dem Bild von vorhin in dieser Aktivität erscheinen. Lade das Programm auf deinen Edison-Roboter herunter. Drücke die Play-Taste (Dreieck) auf deinem Roboter. Dadurch wird das Hauptprogramm gestartet und Edisons LEDs blinken an und aus. Drücke nun den runden Knopf am Roboter. Dadurch wird das Hauptprogramm unterbrochen und die Subroutine ausgeführt. Diese Subroutine sagt Edison, dass er 3 Sekunden warten soll und dann zum Hauptprogramm zurückkehren soll.

Du kannst dieses Programm benutzen, um deinen Edison-Roboter in einen Decider-Bot zu verwandeln!

Denke dir eine Frage aus, die du mit einem 'Ja' oder 'Nein' beantworten kannst. Der Entscheidungsroboter wird dir helfen, diese Frage zu beantworten. Wenn die rechte LED leuchtet, wenn du den runden Knopf drückst, ist die Antwort auf deine Frage 'ja', aber wenn die linke LED leuchtet, dann ist die Antwort 'nein'.

Woran liegt das?

Denkt daran, dass ein Interrupt das Hauptprogramm sofort pausiert, so dass es möglich ist, dass keine der beiden LEDs aufleuchtet, wenn die Subroutine läuft. Wenn das Hauptprogramm die rechte LED ausgeschaltet hat, aber noch nicht die linke LED angeschaltet hat, wenn der Interrupt auftritt, dann sind beide LEDs aus. Das ist ein bisschen so, als würde man eine Münze werfen und sie auf ihrem Rand landen lassen - selten, aber es kann passieren!

Name_____

U3-2.1a:Versuche es mit Klatschen

Erinnerst du dich an Edisons Schallsensor? Das ist der besondere Teil der Technik, der sowohl ein Summer als auch ein Tonsensor ist. Das ist der Teil von Edison, der Geräusche macht, wie Pieptöne oder Musiknoten, aber auch Geräusche erkennen kann, wie ein Klatschen.

Du kannst Edisons Geräuschsensor benutzen, um einen Interrupt in einem Programm auszulösen. Auf diese Weise kannst du einen Deciderbot machen, der auf das Geräusch eines 171geräusches reagiert!

Was zu tun ist

Schreibe ein Programm, um Edison in einen Decider Bot zu verwandeln. Dein Hauptprogramm sollte Edison seine beiden LEDs für immer an- und ausschalten lassen. Du brauchst auch eine Subroutine, die das Hauptprogramm unterbricht, wenn der Roboter einen Tripper entdeckt. Deine Subroutine sollte Edison anweisen, auf ein paar Geräusche zu warten, damit du sehen kannst, welche LED an ist und deine Antwort erhältst.

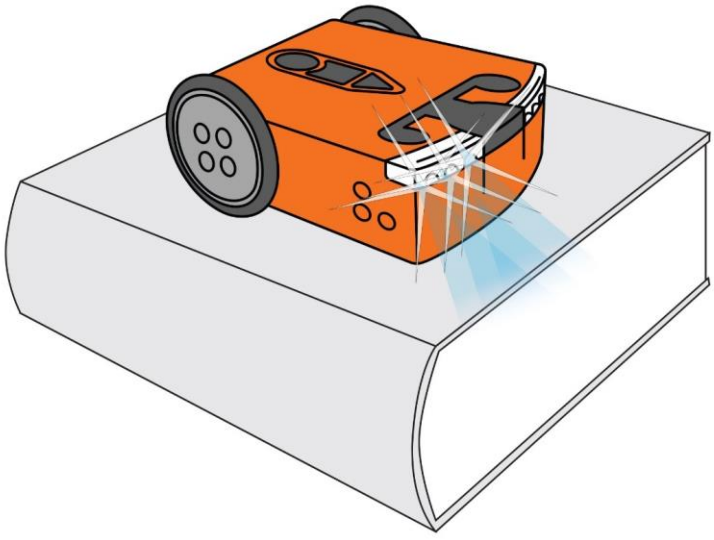
Nicht vergessen

In EdScratch musst du einen Block aus der Kategorie "Ereignisse" verwenden, um als Interrupt zu fungieren und der erste Block am Anfang jeder Subroutine zu sein.

Lade dein Programm auf deinen Edison-Roboter herunter und teste es aus.

Tipp!

Du kannst das Decider Bot Programm aus Aktivität U3-2.1 als Basis für dein Programm verwenden.



Name_____

U3-2.1b: Betrüger-Bot

Einen Decider-Bot zu benutzen ist eine faire Möglichkeit, zwischen zwei Optionen zu wählen. Die LEDs blinken so schnell, dass es sehr schwer ist, zu 'schummeln' und den Roboter dazu zu bringen, so zu antworten, wie du willst. Ihr könnt einen Decider-Bot mit einem 'Cheat' bauen... aber dazu müsst ihr eine zweite Subroutine hinzufügen!

Was zu tun ist

Schreibe ein Programm, das Edison in einen Decider-Bot mit einem geheimen Cheat verwandelt. Dein Hauptprogramm sollte Edison seine LEDs für immer an- und ausschalten lassen. Außerdem brauchst du zwei Unterprogramme: ein Unterprogramm, das fair ist und eines, das schummelt. Die 'faire' Subroutine muss das Hauptprogramm unterbrechen, wenn der Roboter erkennt, dass eine Taste gedrückt wird, und Edison sagen, dass er auf ein paar Töne warten soll, damit du sehen kannst, welche LED an ist, um deine Antwort zu erhalten.

Nicht vergessen

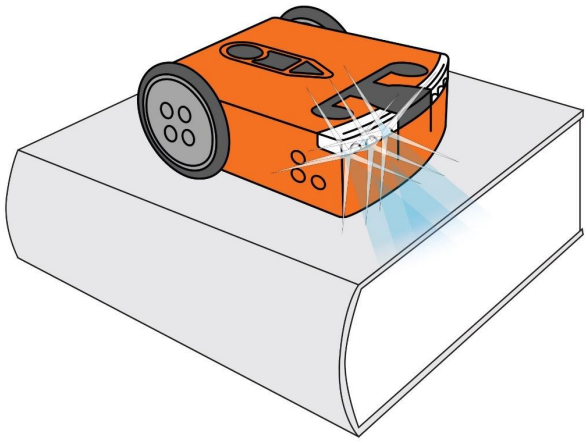
In EdScratch musst du einen Block aus der Kategorie "Ereignisse" verwenden, um als Interrupt zu fungieren und der erste Block am Anfang jeder Subroutine zu sein.

Die zweite Subroutine sollte auch das Hauptprogramm unterbrechen, wenn der Roboter erkennt, dass eine andere Taste gedrückt wurde. Anstatt aber einfach nur zu warten, musst du diese Subroutine so gestalten, dass sie dir eine bestimmte Antwort gibt.

Lade dein Programm auf deinen Edison-Roboter herunter und teste es aus.

Tipp!

Du kannst das Decider Bot Programm aus Aktivität U3-2.1 als Basis für dein Programm verwenden.



Name_____

U3-2.1c: Wähle!

Ein Decider-Bot wählt eine Antwort aus zwei verschiedenen Optionen aus. Wenn du Edison als Decider-Bot mit seinen LEDs verwendest, musst du dir merken, für welche Auswahl die rechte LED steht und für welche die linke LED. Anstatt sich daran zu erinnern, warum nicht eine Möglichkeit schaffen, dass Edison die Antwort beleuchtet!

Was zu tun ist

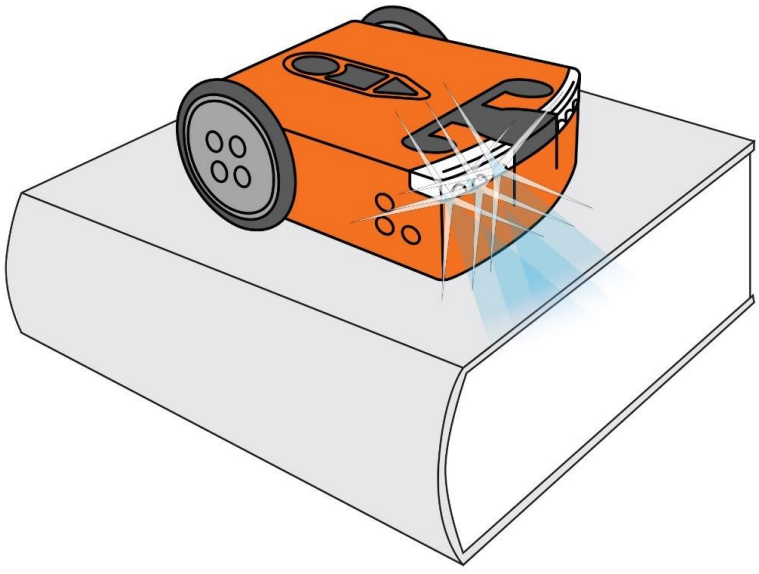
Schreibe ein Programm, um Edison in einen Decider Bot zu verwandeln. Dein Hauptprogramm sollte Edison seine LEDs für immer an- und ausschalten lassen. Du brauchst auch eine Subroutine, die das Hauptprogramm unterbricht, wenn der Roboter erkennt, dass ein Knopf gedrückt wird, und Edison anweist, ein paar Sekunden zu warten, damit du sehen kannst, welche LED an ist und deine Antwort erhältst.

Du musst auch eine Art physisches Setup erstellen, damit die Wahl des Entscheidungsroboters die Antwort aufleuchten lässt. Auf diese Weise kannst du zwei Auswahlmöglichkeiten aufschreiben, und Edison wird eine aufleuchten lassen!

Lade dein Programm auf deinen Edison-Roboter herunter und teste es mit deiner Kreation.

Tipp!

Du kannst das Decider Bot Programm aus Aktivität U3-2.1 als Basis für dein Programm verwenden.



Name_____

U3-2.2: Die Kommentarfunktion

Ein Teil des Erlernens von Code ist das Erlernen einer anderen Sprache: einer Computerprogrammiersprache! Je mehr du in einer Programmiersprache kodierst, desto einfacher ist es, Programme zu verstehen, die in dieser Sprache geschrieben wurden. Du kannst dir ein Programm ansehen, jedem Befehl der Reihe nach folgen und anfangen herauszufinden, was das Programm tun wird, wenn du es ausführst.

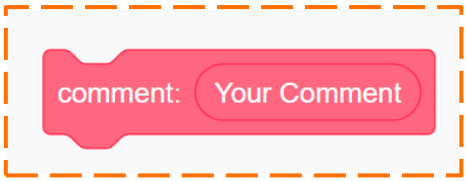
Es gibt auch ein Werkzeug, das es uns erleichtert, Programme zu lesen, die „comments“ - Kommentare genannt werden.

Fachjargon

Beim Programmieren sind „comments" Anmerkungen, die der Programmierer hinzufügt, um den Überblick zu behalten. Comments sind Botschaften, um zu dokumentieren, was im Programm passiert und Dinge zu klären, damit die Leute das Programm verstehen können.

Wie Kommentare aussehen, hängt von der Programmiersprache ab. Manchmal können Kommentare ein bisschen wie Code aussehen, aber Kommentare sind nicht wirklich Code. Wenn ein Computer ein Programm ausführt, das Kommentare enthält, ignoriert der Computer diese Kommentare. Kommentare sind nur für Leute!

In EdScratch kannst du einen Kommentar zu deinem Programm hinzufügen, indem du den speziellen Block in der Kategorie "comments" benutzt. So sieht der Kommentarblock aus:



Der Kommentarblock sieht ähnlich aus wie andere Blöcke in EdScratch, aber er funktioniert etwas anders. Siehst du, wo in dem Block ‚Your Comment' steht? Dort kannst du in deine Notiz schreiben. Denke daran, dass dieser Block kein Code für Edison ist. Wenn Edison einen Kommentar-Block in einem Programm sieht, überspringt es den Block einfach und geht zum nächsten Befehl über. Du kannst dir die Nachricht, die du in einen Kommentarblock schreibst, als den Eingabeparameter dieses Blocks vorstellen, aber anstatt eine Information für Edison zu sein, ist es eine Notiz für eine Person.

Name_____

Wie benutzt man Kommentare?

In vielerlei Hinsicht hängt es von dir ab, wie du Kommentare in deinem Programm verwendest. Kommentare müssen nicht der Syntax folgen, es gibt also keine bestimmte Art und Weise, wie du deine Kommentare schreiben musst. Du kannst die Dinge in deinen Kommentaren so formulieren, wie du es am sinnvollsten findest. Du kannst auch Kommentare zu deinem Code hinzufügen, wo immer du denkst, dass du eine Notiz brauchst, um zu erklären, was Als Nächstes kommt.

Woran liegt das?

Da Kommentare nur für Leute geschrieben werden, brauchst du dir keine Sorgen machen, dass der Computer versteht, was du meinst. Deshalb müssen Kommentare auch nicht in der Syntax der Programmiersprache geschrieben werden.

Das Hinzufügen von Kommentaren zu einem Programm macht es für andere Leute einfacher, dein Programm zu lesen, aber die Person, die deine Kommentare am ehesten lesen wird, bist in Zukunft du! Das liegt daran, dass Kommentare ein hilfreiches Werkzeug zur Fehlersuche in deinen Programmen sind.

Indem du einen Kommentar hinzufügst, kannst du dein Denken organisieren und verfolgen, was du versuchst zu tun. Auf diese Weise ist es einfacher, wenn etwas in deinem Programm nicht so funktioniert, wie du es beabsichtigt hast, zurückzugehen und zu sehen, wo das Problem liegen könnte.

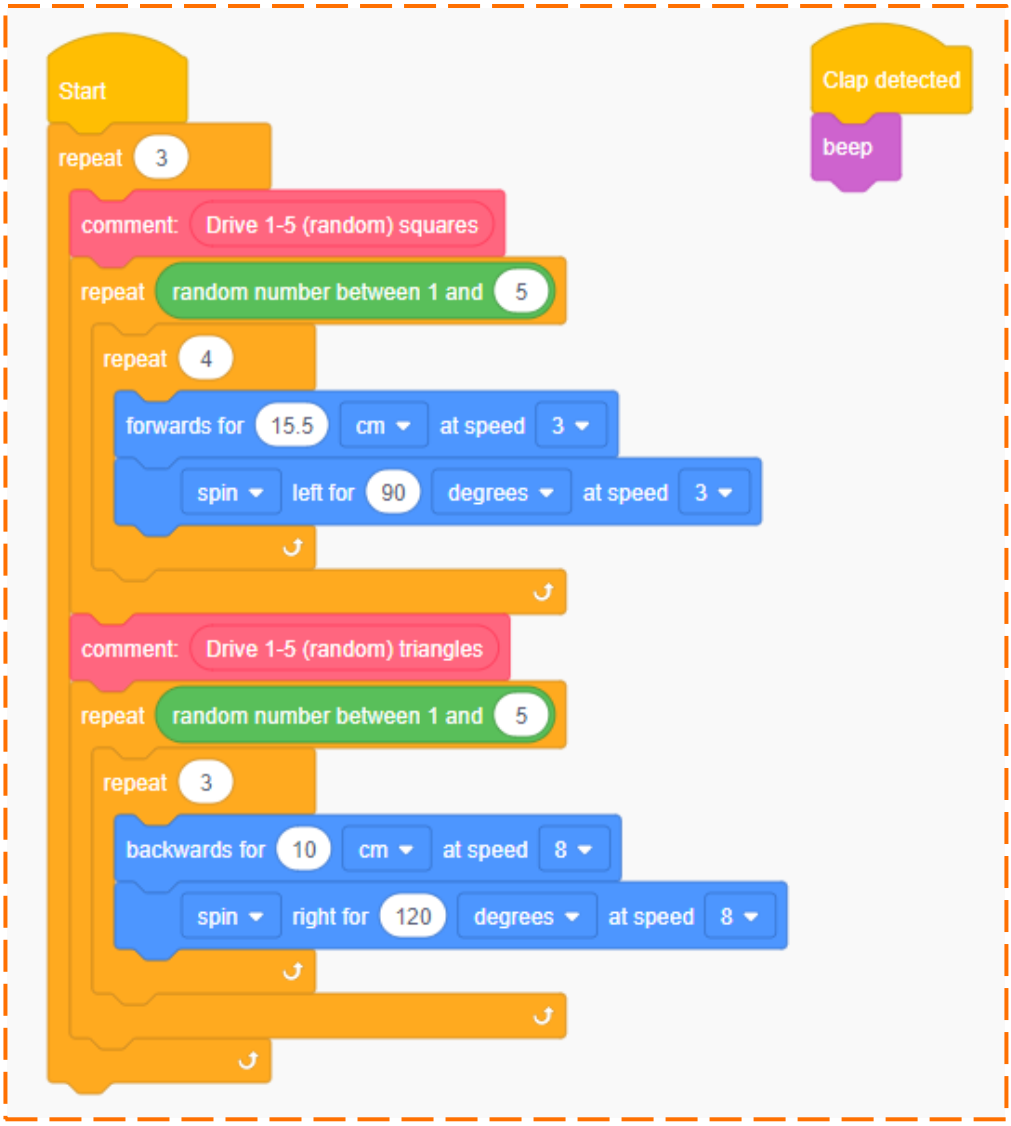
Nicht vergessen

Das Finden und Beheben von Problemen in einem Computerprogramm nennt man Debugging.

Name_____

Probiere es aus!

Schau dir dieses Programm an:



Der Programmierer hat dem Code einige Kommentare hinzugefügt, um das Programm leichter lesbar zu machen. Benutze dieses Bild, um die folgenden Fragen zu beantworten.

Name_____

1. Was denkst du, wird Edison tun, wenn du den Roboter mit dem Code auf dem Bild programmierst?

_____

_____

_____

_____

2. Machen die Kommentare es dir leichter zu verstehen, was der Programmierer will, dass das Programm tut? Warum oder warum nicht?

_____

_____

_____

_____

Versuche nun, deinen Edison mit dem Programm auf dem Bild zu programmieren.

3. Hat das Programm so funktioniert, wie du erwartet hast? Beschreibe alles, was passiert ist, mit dem du nicht gerechnet hast.

_____

_____

_____

_____

Als der Programmierer dieses Programm laufen ließ, passierte etwas Unerwartetes:

"Ich wollte nur, dass Edison piepst, wenn ich klatsche, aber der Roboter piepst die ganze Zeit, während das Programm läuft, immer wieder. Warum ist das so?"

4. Warum piept Edison? *Hinweis*: Gibt es einen Hinweis in der EdScratch-Umgebung?

_____

_____

_____

Name_____

U3-2.2a: Erstellen und kommentieren

Die Verwendung von Kommentaren ist eine gute Möglichkeit, dein Denken zu organisieren, während du Code schreibst. Durch das Hinzufügen von Kommentaren kannst du dir selbst oder jemand anderem eine kleine Nachricht darüber hinterlassen, was der Code macht. Das ist wirklich hilfreich, wenn du später zu einem Programm zurückkehren musst, besonders wenn du etwas debuggen musst. Das Lesen der Kommentare in einem Programm ist ein schneller Weg, um zu wissen, was los ist!

Nicht vergessen

Die Person, die deine Kommentare am ehesten liest, bist du in Zukunft!

Was zu tun ist

In dieser Aktivität musst du ein Programm in EdScratch entwerfen, das mit deinem Edison-Roboter läuft. Was dein Programm macht, ist dir überlassen, aber es muss die folgenden Dinge beinhalten:

- ☐ ein Hauptprogramm
- ☐ mindestens eine Subroutine, die entweder durch ein Knopfdruck- oder ein Klatsch-Ereignis ausgelöst wird
- ☐ mindestens eine Schleife
- ☐ mindestens zwei Arten von Outputs

Entwerfe und schreibe dein Programm in EdScratch. Füge Kommentare hinzu, während du dein Programm schreibst, um den Überblick zu behalten, was du versuchst zu tun und um jemand anderem zu helfen, dein Programm zu lesen.

1. Was soll dein Programm tun? Beschreibe, was dein Programm tun soll, wenn du es in EdScratch laufen lässt.

_____

_____

_____

_____

Name_____

2. Wo in deinem Programm hast du Kommentare eingefügt? Was hast du damit gesagt? Schreibe mindestens ein Beispiel eines Kommentars auf, den du in dein Programm aufgenommen hast. Erkläre, wo du den Kommentarblock eingefügt hast, warum du ihn dort eingefügt hast und was der Kommentar aussagt.

_____

_____

_____

_____

_____

Teste dein Programm mit deinem Edison-Roboter. Funktioniert es so, wie du es erwartet hast? Passiert etwas, was du nicht möchtest? Macht dein Programm alles, was du willst, und geschieht jede Aktion in der Reihenfolge, die du willst?

3. Beschreibe jedes Problem, das du hattest, als du dein Programm zum ersten Mal getestet hast. Was hast du getan, um das Problem zu beheben?

_____

_____

_____

_____

Sobald dein Programm so funktioniert, wie du es willst, geh zurück und schau dir deine Kommentare an. Machen sie alle noch Sinn? Musst du neue Kommentare hinzufügen oder die Kommentare, die du ursprünglich eingegeben hast, wieder loswerden? Verfeinere deine Kommentare, damit sie mit deinem endgültigen Programm funktionieren.

Name_____

4. Wie sieht dein fertiges Programm aus? Schreibe dein Programm unten. Achte darauf, dass du die von dir verwendeten Eingabeparameter und deine Kommentare angibst.

Name_____

U3-2.2b: Teile deine Kommentare

Kommentare helfen dabei, es anderen Leuten leichter zu machen, deinen Code zu lesen und zu verstehen, was dein Programm tun soll. Das heißt, solange andere Leute verstehen, was deine Kommentare bedeuten!

Die meiste Zeit arbeiten die Leute zusammen an Codierungsprojekten, und Kommentare machen es einfacher, zusammenzuarbeiten. Da es aber keine strengen Regeln gibt, wie man Kommentare schreibt, schreiben verschiedene Leute Kommentare auf unterschiedliche Art und Weise. Das Ergebnis ist, dass Kommentare manchmal verwirrender sind als der Code!

Was zu tun ist

Du brauchst mindestens einen Partner für diese Aktivität.

Entwerfe und schreibe ein Programm in EdScratch. Halte dein Programm vor deinem Partner geheim!

Was dein Programm macht, ist dir überlassen. Füge auf jeden Fall Kommentare hinzu, um den Überblick zu behalten, was du versuchst zu tun und um jemand anderem zu helfen, dein Programm zu lesen.

Tipp!

Du kannst die Aktivität U3-2.2a als Inspiration dafür verwenden, was du in dein Programm aufnehmen möchtest.

Wenn sowohl du als auch dein Partner mit dem Schreiben eurer eigenen Programme fertig seid, tauscht euch untereinander aus.

Kannst du das Programm deines Partners verstehen, wenn du seinen Code und seine Kommentare liest, ohne das Programm in Edison laufen zu lassen? Kann euer Partner euer Programm und seine Kommentare verstehen?

1. Gib vor, dass du dafür verantwortlich bist, die Regeln dafür aufzustellen, wie jeder die Kommentare in EdScratch verwenden soll. Welche Regeln sollten alle befolgen, wenn sie Kommentare verwenden, damit die Kommentare hilfreich und leicht zu verstehen sind? Arbeite mit deinem Partner zusammen, um die neuen Regeln für die Verwendung von Kommentaren festzulegen.

_____

_____

_____

_____

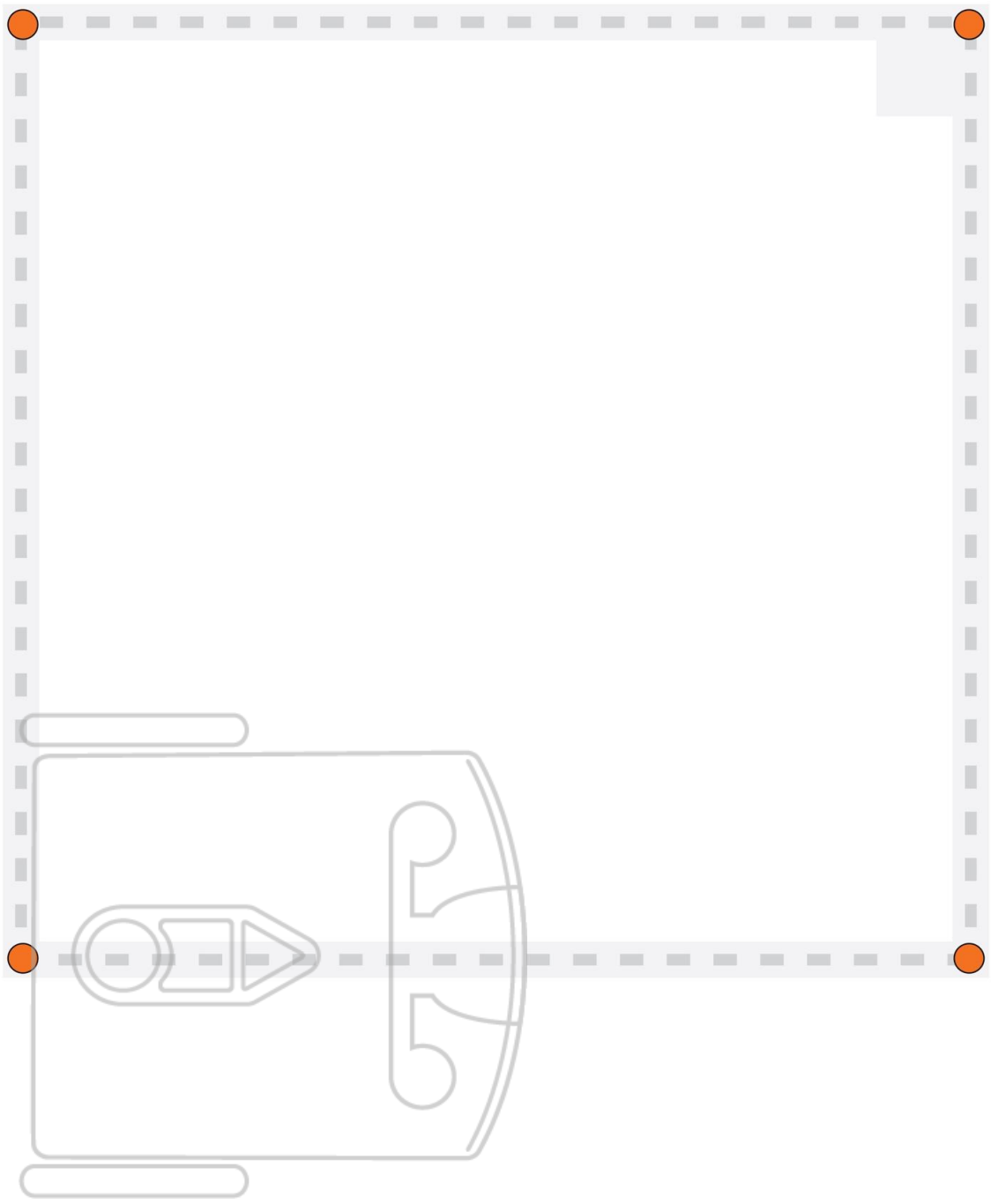
_____

_____

_____

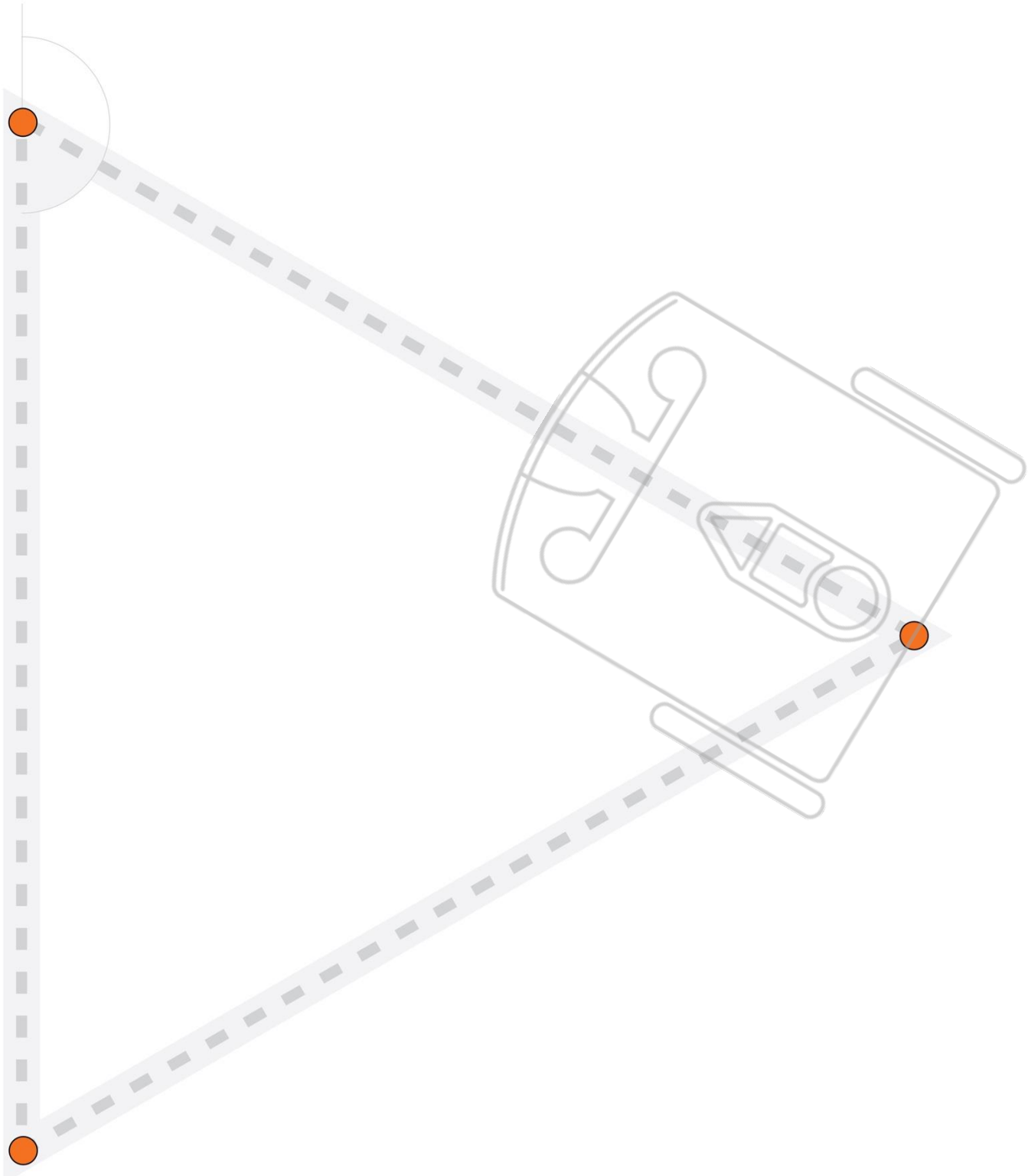
Name_____

Arbeitsblatt U3-1: Ein Quadrat fahren



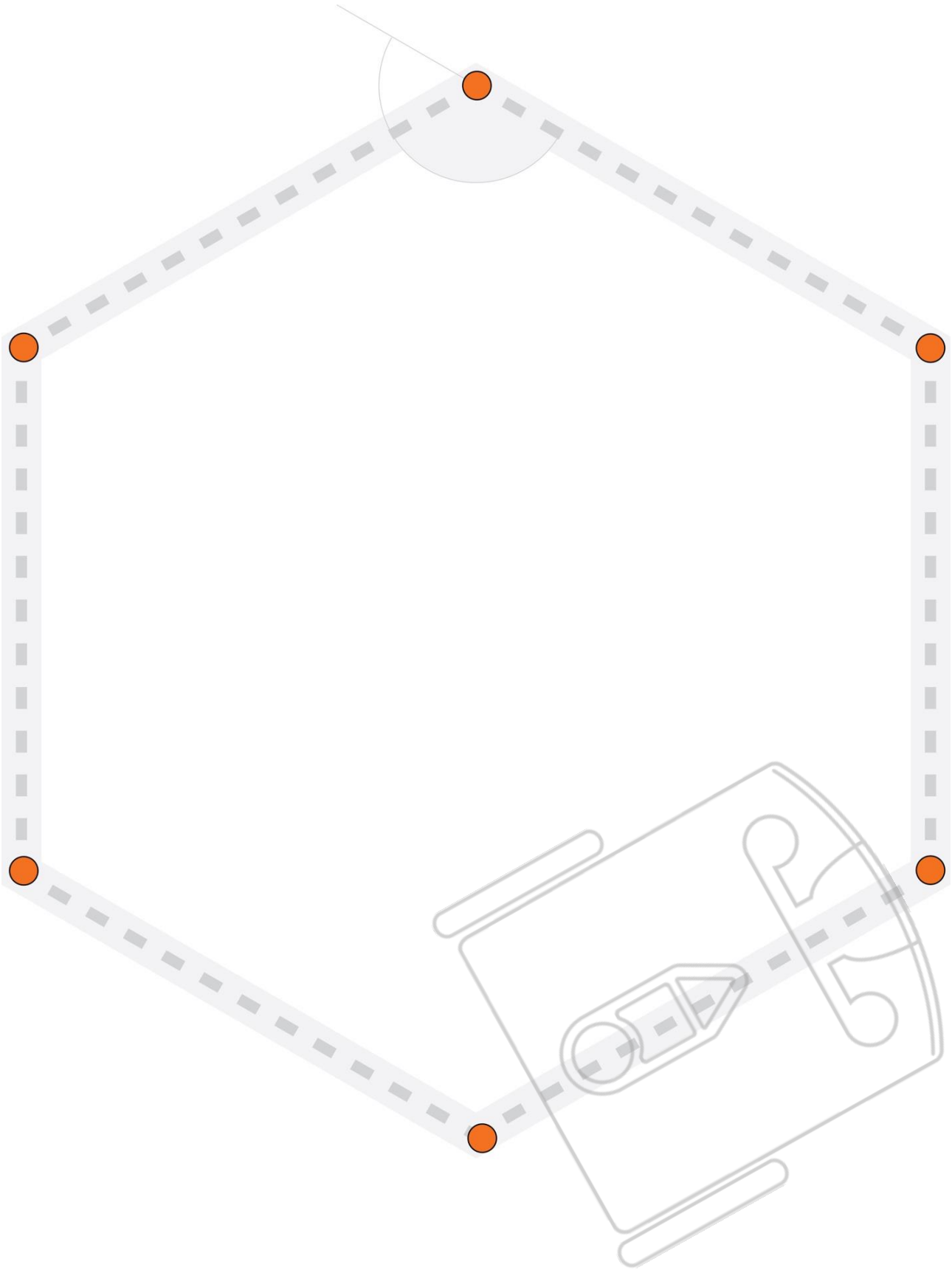
Name_____

Arbeitsblatt U3-2: Ein Dreieck fahren



Name_____

Arbeitsblatt U3-3: Ein Sechseck fahren

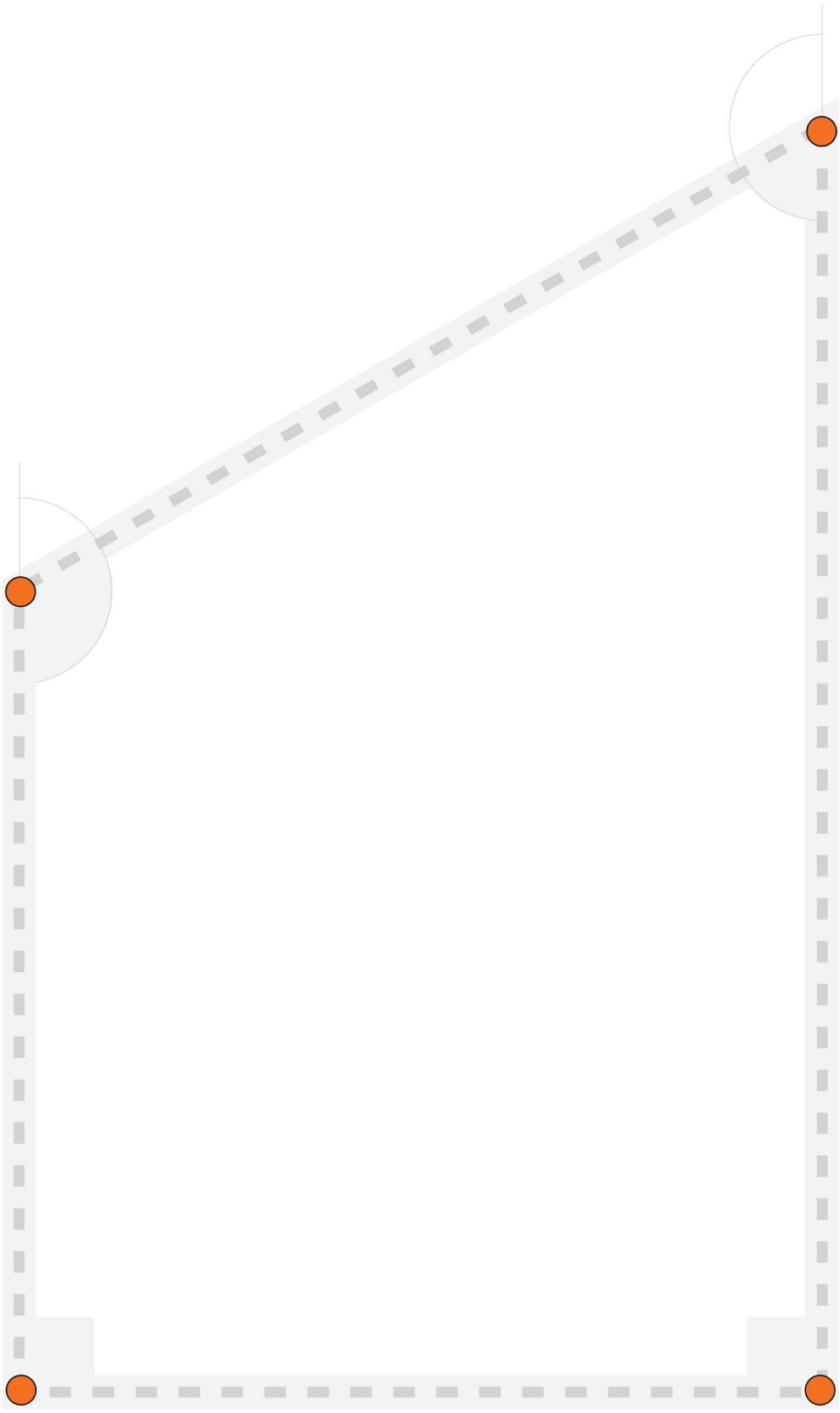


Name_____

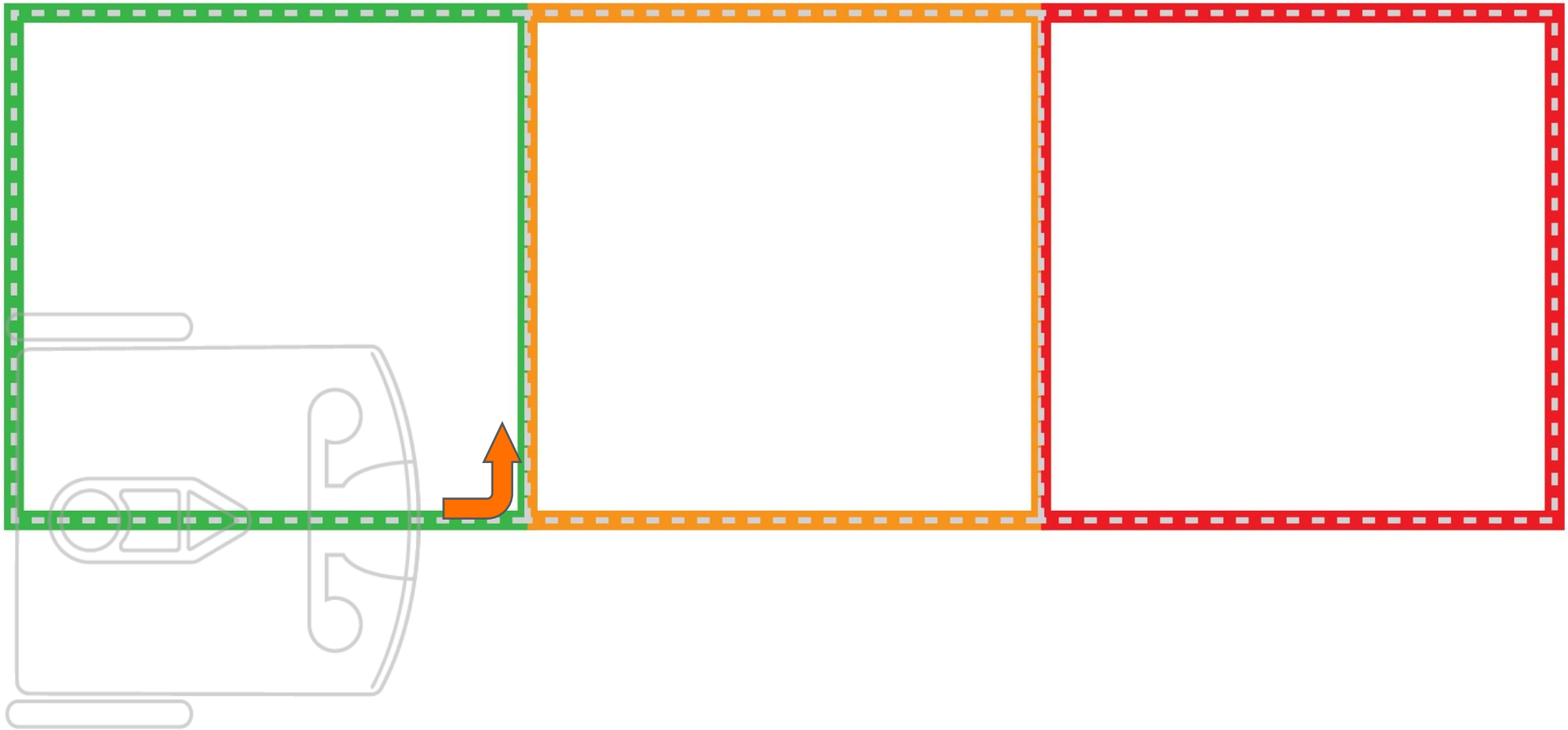
Arbeitsblatt U3-4: Einen Kreis fahren

Name_____

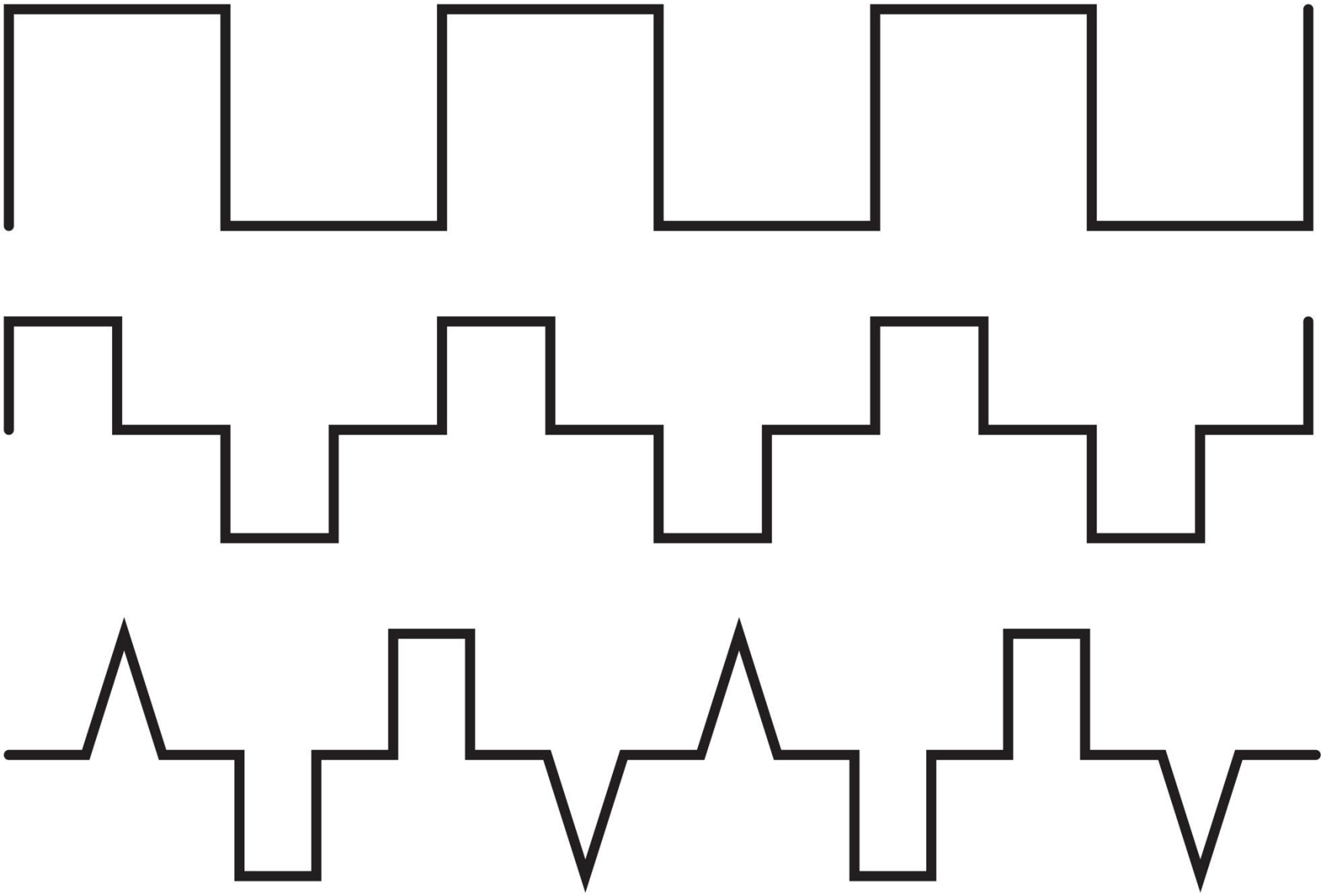
Arbeitsblatt U3-5: Ein Viereck fahren



Arbeitsblatt U3-6: Quadrate wiederholen

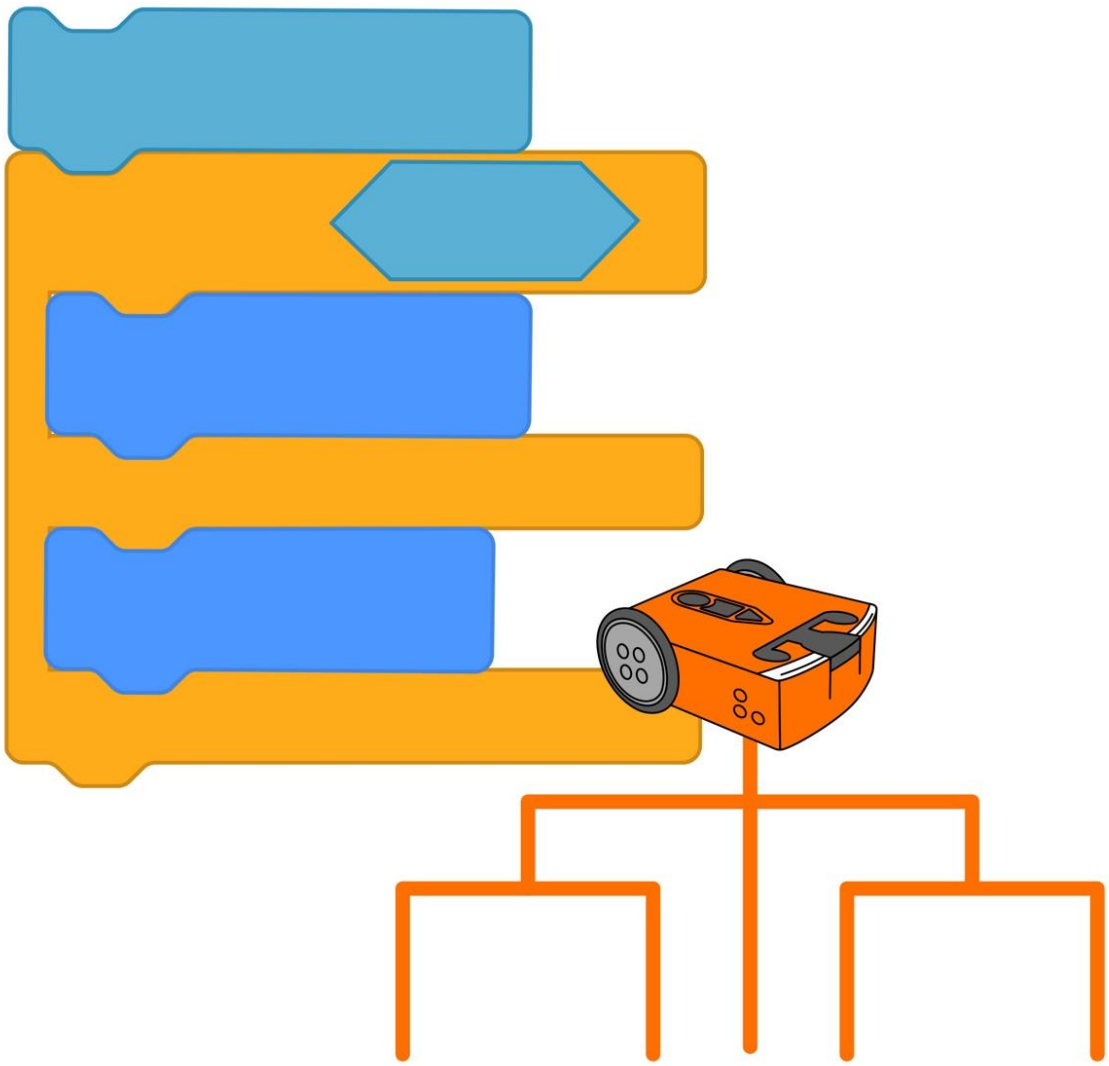


Arbeitsblatt U3-7: Entwürfe zum Fahren



Name_____

Einheit 4: Was wäre, wenn...



Edison

Name_____

U4-1.1: Die Verwendung von Bedingten Anweisungen

Um einen Computer, wie einen Edison-Roboter, dazu zu bringen, das zu tun, was du willst, musst du ihm ganz bestimmte Anweisungen in Form eines Computerprogramms geben. Der Computer folgt dann deinem Code Schritt für Schritt. Er tut alles, was du ihm gesagt hast.

Was ist, wenn du willst, dass der Computer von selbst eine Entscheidung trifft?

Die meisten Computer, auch dein Edison-Roboter, können komplizierte Dinge nicht so entscheiden wie ein Mensch, aber du kannst Edison-Roboter dazu bringen, einfache Entscheidungen zu treffen. Du musst ihm trotzdem genaue Anweisungen geben, die er befolgen muss, damit er weiß, welche Entscheidung er trifft und welche Regeln oder *Bedingungen* für diese Entscheidung gelten. Um diese Art von Programm zu schreiben, musst du eine Art von Programmstruktur verwenden, die als *Bedingte Anweisung (engl.=conditional)* bekannt ist.

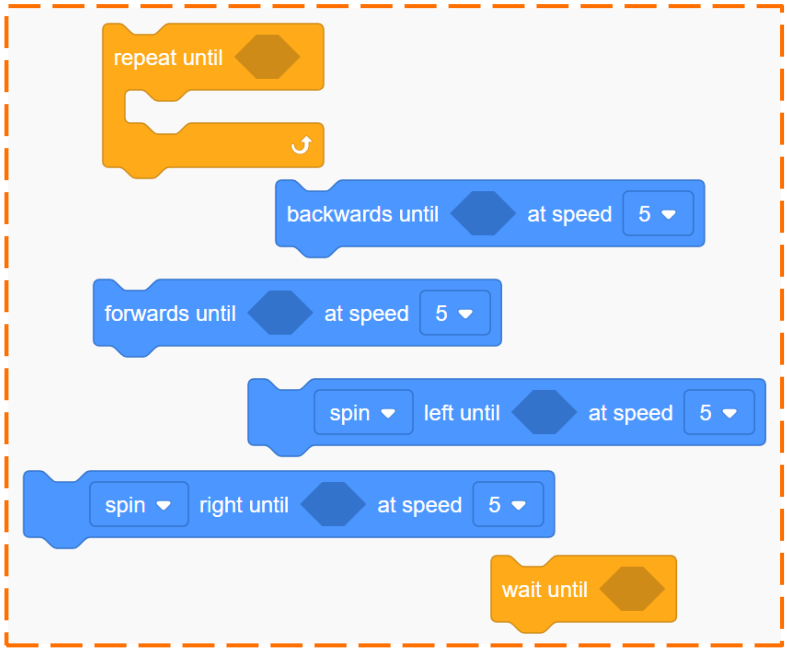
Fachjargon

Eine Bedingung, die manchmal auch bedingte Anweisung genannt wird, ist ein Element des Codes, das von etwas anderem abhängig ist. Dieses Stück Code wird nur dann ausgeführt, wenn seine Bedingung erfüllt ist. Im Code ist eine Bedingung ein vorbestimmter Umstand oder eine Reihe von Faktoren, die erfüllt sein müssen, damit der bedingte Code läuft.

Name_____

Eine Möglichkeit, wie du mit Edison in EdScratch Bedingungen verwenden kannst, ist, indem du ein Programm schreibst, das dem Roboter sagt, dass er etwas tun soll, bis (until) eine Bedingung erfüllt ist.

Schau dir die EdScratch-Blöcke in diesem Bild an:



Alle diese Blöcke sind Conditionals, die die gleiche Formel verwenden, bis die Bedingung (until) erfüllt ist. Jeder Block sagt Edison, dass er eine Aktion ausführen soll, bis eine bestimmte Bedingung erfüllt ist. Aber was ist die Bedingung?

Sieh dir noch einmal die bis-Blöcke auf dem Bild an. Siehst du das diamantförmige Loch in jedem Block? Du gibst einem Till-Block eine Bedingung, indem du einen speziellen Eingabeparameter eingibst.

Nicht vergessen

Es gibt drei Arten von Eingabeparametern in EdScratch:

- Zahlen, die du mit deiner Tastatur in einen Block eingibst,
- Dropdown-Menüs, in denen du eine Option innerhalb des Blocks auswählst, und
- runde oder rautenförmige Löcher, die du mit speziellen Blöcken füllst.

Jeder Eingabeparameter in einem Block gibt Edison eine andere Information, die dein Edison-Roboter benötigt, um diesen Befehl auszuführen. Du kannst dir Eingabeparameter als die Antworten auf Fragen vorstellen, die der Roboter zu dem hat, was du ihm aufträgst.

Name_____

Genau wie jeder andere Code-Block müssen auch Konditionals alle ihre Eingabeparameter ausgefüllt haben, damit sie korrekt funktionieren. Wenn ihr einen der Till-Blöcke in EdScratch benutzt, müsst ihr dem Roboter die Bedingung mit einem rautenförmigen Eingabeparameter geben.

1. Öffne die EdScratch-Programmierumgebung und schau dir die verschiedenen Blöcke an. Welche Blockkategorien enthalten Blöcke, von denen du denkst, dass du einen der Till-Blöcke mit einem Bedingungs-Eingabeparameter versehen kannst? Warum denkst du das?

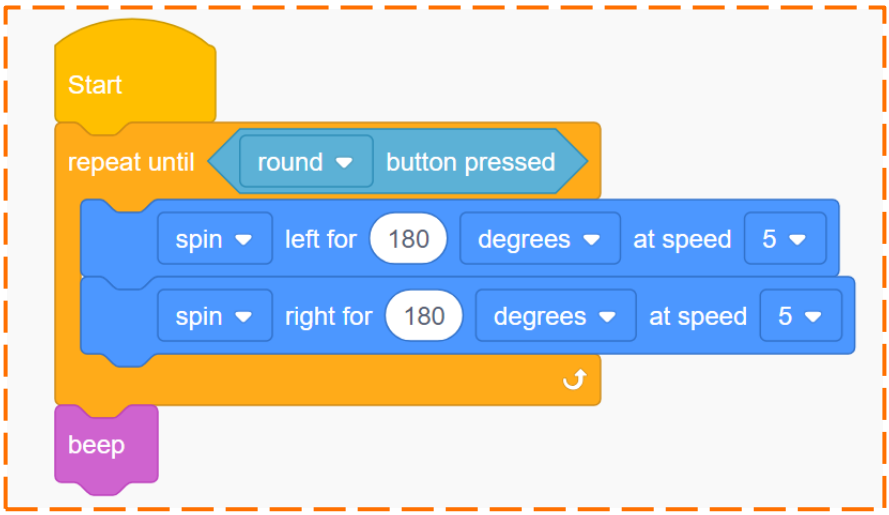
_____

_____

_____

Aufgabe 1: Wiederhole sie bis...

Schau dir dieses EdScratch Programm an:



Dieses Programm benutzt eine repeat until-Schleife, die eine unbestimmte Schleife ist.

Name_____

Woran liegt das?

Jede Schleife, die sich für eine unbestimmte Anzahl von Malen wiederholt, ist eine unbestimmte Schleife.

Der Forever-Block in EdScratch ist ein Beispiel für eine unbestimmte Schleife, weil er sich unendlich oft wiederholt. Der Block „repeat until-Block" ist ein weiteres Beispiel, weil er so lange wiederholt wird, bis seine Bedingung erfüllt ist. Die Bedingung könnte bereits nach einer Schleife erfüllt sein, oder sie könnte nach 20 Schleifen erfüllt sein, oder sie könnte niemals erfüllt sein! Da wir nicht genau wissen, wie oft der Block in einer Schleife wiederholt wird, ist es eine unbestimmte Schleife.

Schreibe das Programm aus dem Bild in EdScratch und lade es auf deinen Edison-Roboter herunter. Starte das Programm und teste es, um zu sehen, wie es funktioniert.

2. Wenn du dieses Programm ausführst, was musst du tun, um Edison zum Piepen zu bringen? Wie kommt das? *Tipp:* schau dir das Programm an und folge jedem Befehl der Reihe nach.

_____

_____

_____

Name_____

Aufgabe 2: Ereignis + Bedingung = Ereignisbedingungen

Eine der Hauptmöglichkeiten, Conditionals in EdScratch zu verwenden, ist, ein Ereignis als Bedingung zu haben. Dies wird Ereignisbedingung genannt.

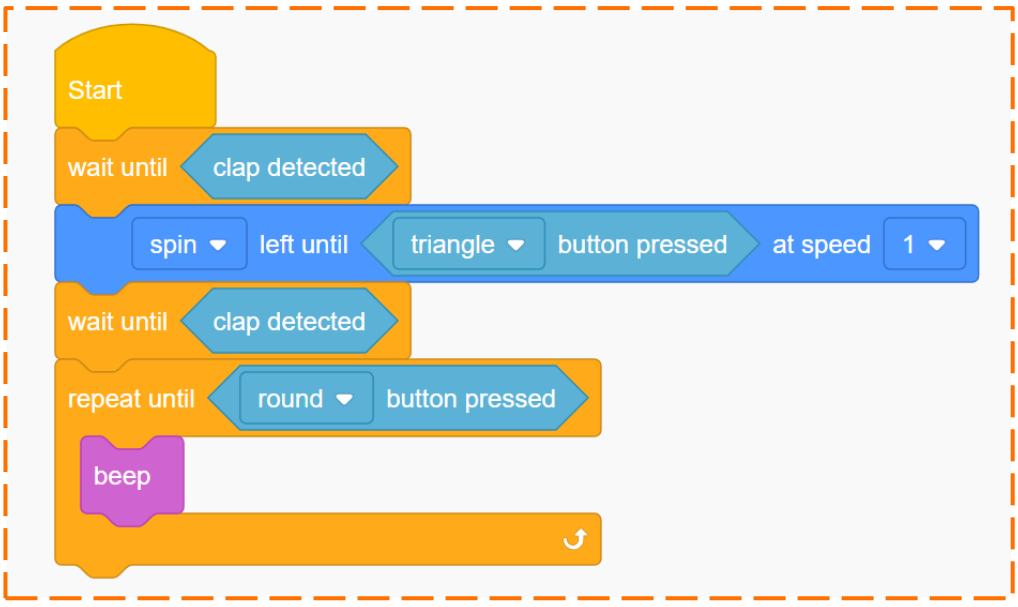
Fachjargon

Erinnere dich daran, dass beim Programmieren ein Ereignis etwas ist, das außerhalb des Programmcodes passiert und das beeinflusst, wie das Programm läuft.

Eine Ereignisbedingung ist eine Bedingung, die ein bestimmtes Ereignis, wie z.B. das Drücken einer Taste, voraussetzt, damit die Bedingung erfüllt wird, wodurch der bedingte Code ausgeführt wird.

Du kannst Ereignisbedingungen mit until-Blöcken in EdScratch verwenden. Edison wird die Aktion des untill-Blocks so lange ausführen, bis das Ereignis eintritt. Sobald das Ereignis eintritt, wird Edison zum nächsten Block im Programm übergehen.

Dieses Programm benutzt eine Menge von until-Blöcken mit Ereignisbedingungen:



Name_____

Schreibe dieses Programm in EdScratch und lade es auf deinen Edison-Roboter herunter. Starte das Programm. Kannst du das Programm dazu bringen, jeden Schritt abzuschließen und erfolgreich zu beenden, so dass der Roboter in den Standby-Modus zurückkehrt?

3. Sobald das Programm läuft, wie viele Ereignisse müssen eintreten, damit das gesamte Programm erfolgreich abgeschlossen werden kann?

   _____

4. Der erste Codeblock in diesem Programm sagt dem Roboter, dass er warten soll, bis ein Klatschen erkannt wird. Fällt dir ein Beispiel für etwas aus dem echten Leben ein, das diese Art von Wartezeit bis zum Bedingungscode verwenden könnte? Was für ein Gerät könnte ein Programm verwenden, das als erste Aktion eine Warte-bis-Bedingung verwendet? Welche Bedingung würde den Bedingungscode auslösen? Was würde das Programm das Gerät tun lassen, wenn die Ereignisbedingung eintritt?

_____

_____

_____

_____

_____

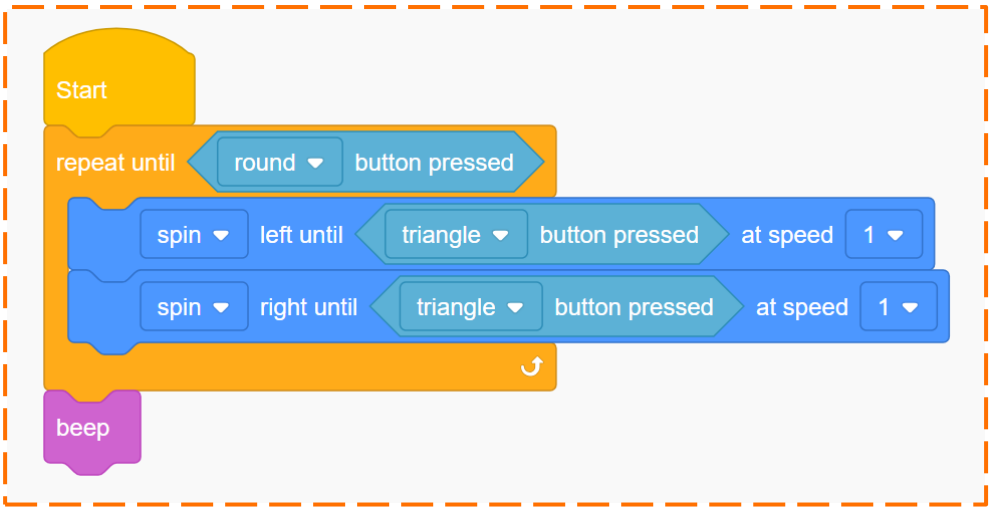
_____

Name_____

U4-1.1a: Roboterfehler oder menschlicher Fehler?

Manchmal, wenn wir ein Programm für Edison schreiben, scheint es, dass der Roboter einfach nicht das macht, was wir wollen. Wie bei allen Computern gibt es Grenzen für das, was Edison-Roboter tun können. Aber bevor du dem Roboter die Schuld dafür gibst, dass dein Programm nicht funktioniert, frage dich, ob das Problem mit dem Roboter zu tun hat... oder mit dem Menschen.

Schau dir dieses Programm an, das drei verschiedene bedingte Anweisungen verwendet:



Dieses Programm bringt Edison nicht dazu, sich so zu verhalten, wie der Programmierer es will:

"Wenn das Programm läuft, wenn ich den runden Knopf drücke, sollte der Roboter meiner Meinung nach aufhören, sich zu bewegen, dann piepen und dann sollte das Programm beendet werden. Das ist nicht das, was passiert! Ich habe auch versucht, zweimal auf die Dreieckstaste zu drücken und dann auf die runde Taste, aber der Roboter dreht sich einfach weiter. Ich glaube, der Roboter muss kaputt sein".

Ist der Roboter wirklich kaputt? Gibt es Fehler in dem Programm? Hat der Programmierer einen logischen Fehler gemacht? Oder ist es etwas anderes?

Name_____

Nicht vergessen

Logische Fehler sind Probleme mit der Logik oder der Denkweise in einem Programm. Wenn ein Programm nicht so funktioniert, wie du es erwartest, kann es sein, dass du einen logischen Fehler hast. Edison führt das Programm vielleicht genau so aus, wie du den Code geschrieben hast, aber deine Art, über das Programm nachzudenken, lässt es so aussehen, als würde es nicht funktionieren.

Denke daran, dass Computer nicht wie ein Mensch denken können. Deshalb musst du computergestütztes Denken benutzen, um zu planen, Probleme zu lösen und Informationen zu analysieren, genau wie ein Computer.

Was ist zu tun?

Mal sehen, ob wir helfen können, herauszufinden, was mit dem Programm los ist und es dem Programmierer erklären können.

Als erstes musst du das Programm für dich selbst ausführen. Auf diese Weise kannst du sehen, wie es funktioniert und anfangen, nach Fehlern zu suchen. Wenn du das Programm ausführst, kannst du auch das Denken des Programmierers überprüfen, um zu sehen, ob er logische Fehler gemacht hat.

Nicht vergessen

Um das Programm zum Laufen zu bringen, drücke einmal die 'Play'-Taste (Dreieck). Dies startet nur das Programm, es zählt nicht als ein Knopfdruck innerhalb des Programms.

Schreibe das Programm in EdScratch und lade es auf deinen Edison-Roboter herunter. Dieses Programm verwendet Ereignisbedingungen, also musst du das Programm starten und es dann testen, um zu sehen, ob sich der Roboter bei jedem Ereignis so verhält, wie du es erwartest.

1. Bringe das Programm zum Laufen, indem du den 'play' (Dreieck) Knopf drückst. Was bringt das den Roboter dazu zu tun?

_____

2. Ist das das Verhalten, das du erwartet hast? Warum oder warum nicht?

_____

_____

Name_____

3. Drücke nun die Dreieckstaste. Der Roboter beginnt sich nach rechts zu drehen. Warum passiert das?

_____

_____

Der Programmierer sagte, dass es zwei Probleme damit gab, wie dieses Programm funktioniert:

Problem #1:

"Ich habe auch versucht, zweimal auf die Dreieckstaste zu drücken und dann auf die runde Taste, aber der Roboter dreht sich einfach weiter".

Problem #2:

"Wenn das Programm läuft, wenn ich den runden Knopf drücke, sollte der Roboter meiner Meinung nach aufhören, sich zu bewegen, dann piepen und dann sollte das Programm beendet werden. Das ist nicht das, was passiert!"

Lass das Programm noch einmal in deinem Edison-Roboter laufen. Folge den Schritten, die der Programmierer beschrieben hat, um zu sehen, ob du die Probleme, die der Programmierer erlebt hat, reproduzieren kannst. Hast du die gleichen Probleme wie der Programmierer? Denkst du, dass die Probleme menschliche Probleme oder Roboterprobleme sind?

Was geht hier vor sich?

Schauen wir uns die beiden Probleme nacheinander an.

Problem #1:

"Wenn das Programm läuft, wenn ich den runden Knopf drücke, sollte der Roboter meiner Meinung nach aufhören, sich zu bewegen, dann piepen und dann sollte das Programm beendet werden. Das ist nicht das, was passiert!"

Dies ist ein menschliches Problem. Der Programmierer hat einen logischen Fehler gemacht, als er darüber nachdachte, wie das Programm Edison dazu bringen sollte, sich zu verhalten.

Name_____

Um den Fehler zu verstehen, versuche noch einmal, das Programm in deinem Edison laufen zu lassen. Dieses Mal, wenn das Programm läuft, drücke den runden Knopf einmal. Dann drücke die Dreieckstaste zweimal.

Der Roboter dreht sich nach links, dann nach rechts, dann piepst er, und das Programm wird beendet.

Woran liegt das?

Denke daran, dass der Block "Wiederholen, bis die Bedingung erfüllt ist" (repeat until condition) eine unbestimmte Schleife ist, die so lange wiederholt wird, bis die Bedingung erfüllt ist.

Die Schleife sagt Edison, dass er jedes Element des Codes innerhalb der Schleife der Reihe nach ausführen soll und dann wieder an den Anfang der Schleife zurückkehren soll. Wenn die Bedingung der Schleife NICHT erfüllt ist, dann sagt die Schleife Edison, dass er den Code innerhalb der Schleife erneut starten soll. Du kannst dir das so vorstellen wie die Schleife, die Edison die Frage stellt: 'Ist der runde Knopf gedrückt worden? Wenn die Antwort 'nein' ist, dann schickt die Schleife Edison zurück in die Schleife. Wenn die Antwort 'ja' ist, dann schickt die Schleife Edison zu dem Code, der stattdessen nach der Schleife kommt.

Ein Programm wird nur prüfen, ob die Bedingung der Schleife am Anfang der Schleife erfüllt ist, nicht während der Code innerhalb der Schleife läuft. Der Roboter muss zuerst den gesamten Code innerhalb der Schleife vervollständigen, dann geht er zurück an den Anfang der Schleife und überprüft die Bedingung.

Unser Programmierer hat den runden Knopf gedrückt, hat aber die Bedingungen in den Codeblöcken innerhalb der Schleife nicht erfüllt. Das hat den Fehler verursacht!

4. Erkläre mit deinen eigenen Worten den logischen Fehler, den der Programmierer gemacht hat, der dieses Programm geschrieben hat.

_____

_____

_____

_____

_____

Name_____

Schauen wir uns nun das zweite Problem an:

Problem #2:

"Ich habe auch versucht, zweimal auf die Dreieckstaste zu drücken und dann auf die runde Taste, aber der Roboter dreht sich einfach weiter".

Auch dies ist ein menschliches Problem, aber es ist nicht gerade ein logischer Fehler. Das Problem hier ist, dass Menschen, verglichen mit einem Roboter, etwas langsam sind. Das liegt daran, dass Roboter sich sehr, sehr schnell durch den Code bewegen!

Woran liegt das?

Denke daran, dass ein Programm nur prüft, ob die Bedingung der Schleife am Anfang der Schleife erfüllt ist.

In diesem Programm wird, sobald die Dreieckstaste zum zweiten Mal gedrückt wird, der Block nach rechts gedreht, bis die Dreieckstaste gedrückt wird und der Code an den Anfang der Schleife springt, um zu prüfen, ob die runde Taste gedrückt wurde.

Dies geschieht sehr schnell. Es dauert weniger als 10 Millisekunden, bis der Code prüft, ob die runde Taste gedrückt wurde. Das ist weniger als 1/100stel Sekunde!

Unser Programmierer hat zwar den runden Knopf gedrückt, aber bis dahin hatte der Code bereits die Bedingung für die Schleife überprüft und den Roboter zurück in die Schleife geschickt. Der Programmierer kam zu spät!

5. Dinge, die nicht so funktionieren, wie wir erwarten, dass sie funktionieren, passieren ständig im Code. Das kann sehr frustrierend sein! Denk darüber nach, was du das nächste Mal tun kannst, wenn du ein Programm schreibst, das nicht zu funktionieren scheint. Schreibe eine Nachricht an dein Zukunfts-Selbst. Mache ein paar Vorschläge, was du tun kannst, um zu versuchen, das Problem zu beheben.

_____

_____

_____

Name_____

_____

_____

_____

_____

_____

Name_____

U4-1.2: Die „Wenn - Dann“ - Funktion (if-statements)

Durch die Verwendung von Conditionals im Coding kannst du Programme schreiben, die den Computer bitten, etwas zu entscheiden. Mit Conditionals gibst du dem Computer genaue Anweisungen, die er befolgen soll, damit er weiß, welche Entscheidung er trifft und unter welchen Bedingungen er diese Entscheidung trifft.

In der Programmierung ist der gebräuchlichste Weg, Bedingungen für einen Computer zu setzen, die Verwendung einer if-Anweisung.

Fachjargon

Eine if-Anweisung ist eine bedingte Anweisung. Es ist ein Element des Codes, das von etwas anderem abhängig ist. Dieser Teil des Codes wird nur passieren, wenn die Bedingung erfüllt ist. Deshalb nennt man es eine 'if'-Anweisung!

Wahrscheinlich benutzt man 'if'-Anweisungen, um im Leben ständig Entscheidungen zu treffen, vielleicht ohne es zu merken. Schau dir diese Beispiele an:

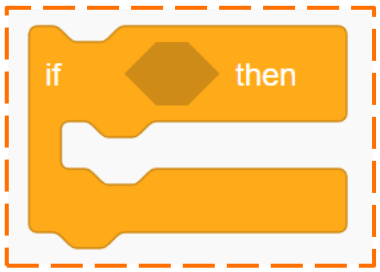
- WENN es draußen kalt ist, DANN ziehe ich eine Jacke an, bevor ich das Haus verlasse.
- WENN ich nach der Schule hungrig bin, DANN esse ich einen Snack.

1. Denke über eine Bedingung nach, der du in deinem täglichen Leben begegnest. Schreibe es mit der Formel 'wenn__, dann___'.

IF _____

THEN _____

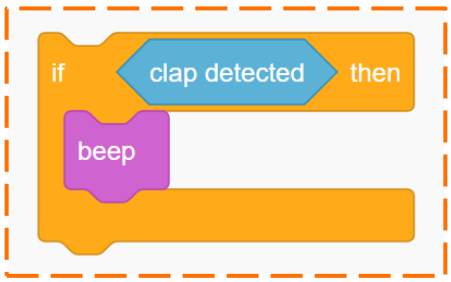
Im Code folgen die 'wenn'-Aussagen derselben Formel. Schau dir den if-Block von EdScratch an:



Name_____

Siehst du die 'wenn__, dann___'-Formel in dem Block? Wenn du eine 'if'-Anweisung im Code verwendest, sagst du dem Computer, dass WENN die 'if'-Bedingung eintritt, DANN machst du die bedingte Aktion.

Zum Beispiel, WENN Klatschen entdeckt wird, DANN piepst es:



Du kannst dem Computer auch sagen, was er tun soll, wenn die Bedingung NICHT eintritt. Um dies zu tun, musst du eine Art von Bedingung verwenden, die man Wenn-Ansonsten-Anweisung nennt.

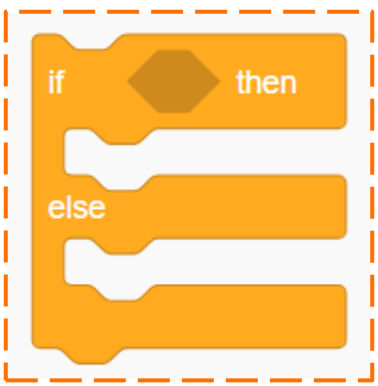
Fachjargon

Eine Wenn-Ansonsten-Anweisung (if-else) ist eine bedingte Anweisung. Genau wie alle Bedingungen ist sie ein Element des Codes, das von etwas anderem abhängig ist. Eine bedingte Anweisung sagt dem Programm, was zu tun ist, wenn die Bedingung erfüllt ist, und sagt dem Programm auch, was zu tun ist, wenn die Bedingung NICHT erfüllt ist.

Der 'else'-Teil der Wenn-Ansonsten-Anweisung sagt dem Programm, was es tun soll, wenn die Bedingung nicht erfüllt ist.

Du kannst dir eine Wenn-Ansonsten-Anweisung wie einen Entscheidungspunkt für das Programm vorstellen. Eine "Wenn-ich-selbst"-Anweisung sagt dem Programm, was es tun soll: "Wenn die Bedingung erfüllt ist, dann tu was A. Wenn die Bedingung nicht erfüllt ist, dann tu was B."

So sieht der Wenn-Ansonsten-Block in EdScratch aus:



Genau wie der Wenn-Dann-Block, verwendet der Wenn-Ansonsten-Block die grundlegende 'wenn__, dann___'-Formel, sagt aber auch, was zu tun ist, wenn die Bedingung nicht erfüllt ist. Eine if-else-Anweisung lässt dich eine bedingte Wahl treffen:

- WENN es draußen kalt ist, DANN ziehe ich eine Jacke an, bevor ich das Haus verlasse. ANSONSTEN gehe ich in einem T-Shirt raus.
- WENN ich nach der Schule hungrig bin, DANN esse ich einen Snack. ANSONSTEN warte ich bis zum Abendessen, um zu essen.

Die Verwendung einer if-else-Anweisung zwingt ein Programm zu verzweigen. Es wird entweder einen Weg einschlagen oder einen anderen Weg einschlagen.

Probiere es aus!

Lasst uns üben, die 'if__, then___ else____'-Formel zu benutzen, um zu sehen, wie es Programme zum Verzweigen bringt. Für diese Aktivität musst du das Arbeitsblatt U4-1 verwenden. Dieses Arbeitsblatt hat eine spezielle Schatzkarte, die nur mit if-else-Aussagen gelöst werden kann.

Folge den Anweisungen, um die Lage der einzelnen Schätze zu bestimmen.

Tipp!

Wusstest du, dass einige mathematische Symbole auch in der Kodierung verwendet werden? Für diese Aktivität müsst ihr diese Symbole verwenden:

A = B bedeutet 'A ist dasselbe wie B'.

A > B bedeutet 'A ist größer als B'.

A < B bedeutet 'A ist kleiner als B'.

Name_____

2. Die Kugel von Marshmallow

Beginne am Startplatz.

Wiederhole 3 Mal:

WENN die Zahl < 7, DANN geh nach links. ANSONSTEN nach rechts gehen.

Wo ist die Kugel von Marshmallow? _____

3. Der Umhang der Pfannkuchen

Beginne am Startplatz.

Wiederhole 3 Mal:

WENN die Zahl < 2, DANN geh nach links. ANSONSTEN nach rechts gehen.

Wo ist der Umhang der Pfannkuchen? _____

4. Das Omelett der Weltraum-Eier

Beginne am Startplatz.

Wiederhole 3 Mal:

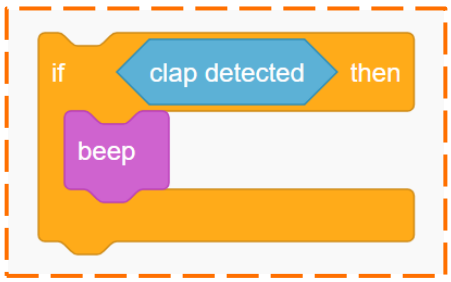
WENN die Zahl < 7, DANN geh nach links. ANSONSTEN nach rechts gehen.

Wo ist das Omelett aus Weltraumeiern? _____

Name_____

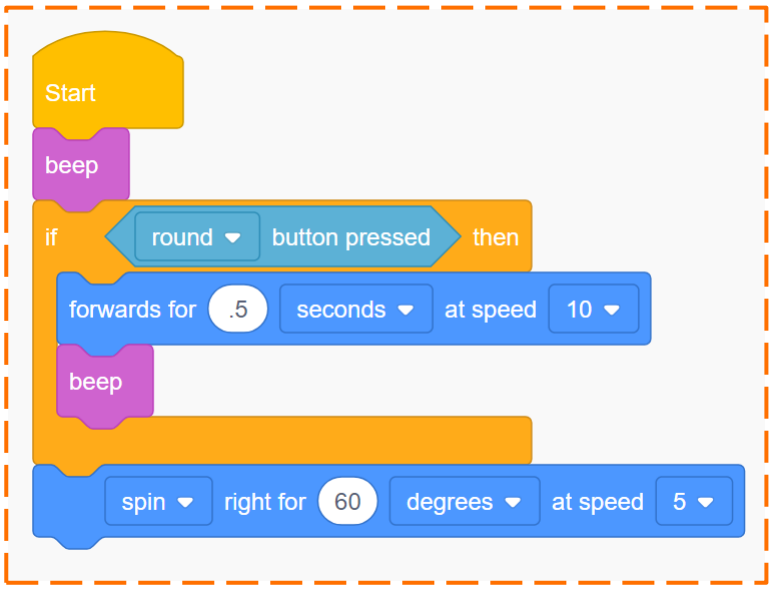
U4-1.3: Wenn-Aussagen und Sequenzen

Wenn du eine 'if'-Anweisung im Code verwendest, sagst du dem Computer, was er tun soll, wenn dieser Zustand eintritt. In EdScratch benötigt der if-Block, dass du die Bedingung mit einem rautenförmigen Eingabeparameter angibst. Du sagst dem Roboter auch, was die bedingte Aktion ist, indem du einen oder mehrere Blöcke in den 'Mund' des if-Blocks legst:



Was passiert in einem Programm, das einen if-Block benutzt, wenn die Bedingung nicht erfüllt ist?

Schau dir das folgende Programm an:



1. Was muss in diesem Programm geschehen, damit die Bedingung im 'if'-Block erfüllt wird?

_____

_____

Schreibe das Programm in EdScratch. Lade das Programm herunter und führe es in deinem Edison-Roboter aus.

Name_____

2. Was ist passiert, als du dieses Programm ausgeführt hast? Wurde der bedingte Code (der Code innerhalb des if-Blocks) ausgeführt?

_____

_____

3. Von dem, was du gefunden hast, was glaubst du, was in einem Programm passiert, das einen if-Block benutzt, wenn die Bedingung NICHT erfüllt ist?

_____

_____

_____

Woran liegt das?

Denke daran, dass alle Programme den Code Schritt für Schritt der Reihe nach durchlaufen. Wenn das Programm zu einem if-Block gelangt, prüft es, ob die Bedingung erfüllt ist. Wenn das der Fall ist, führt das Programm den Code innerhalb des Blocks aus. Wenn die Bedingung nicht erfüllt ist, überspringt das Programm den Code im if-Block und geht zur nächsten Codezeile im Programm über.

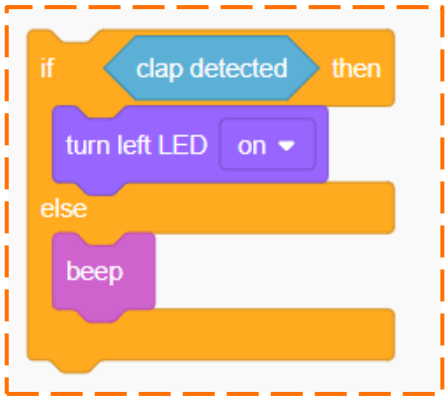
Edison geht sehr schnell von Code-Block zu Code-Block weiter. Es dauert weniger als 10 Millisekunden, bis der Roboter bereits im if-Block ist und auf den runden Knopf drückt. Das ist weniger als 1/100stel Sekunde! Es ist fast unmöglich, den runden Knopf rechtzeitig zu drücken.

Wenn ihr sehen wollt, wie der bedingte Code läuft, welchen zusätzlichen Block könntet ihr dem Programm hinzufügen, um euch mehr Zeit zu geben, den runden Knopf zu drücken?

Genau wie bei einem Wenn-Block, wenn ein EdScratch Programm zu einem Wenn-Ansonsten-Block gelangt, prüft es, ob die Bedingung erfüllt ist. Ein "Wenn-Selbst"-Block sagt dem Roboter, was er tun soll, wenn die Bedingung erfüllt ist und was er tun soll, wenn die Bedingung nicht erfüllt ist.

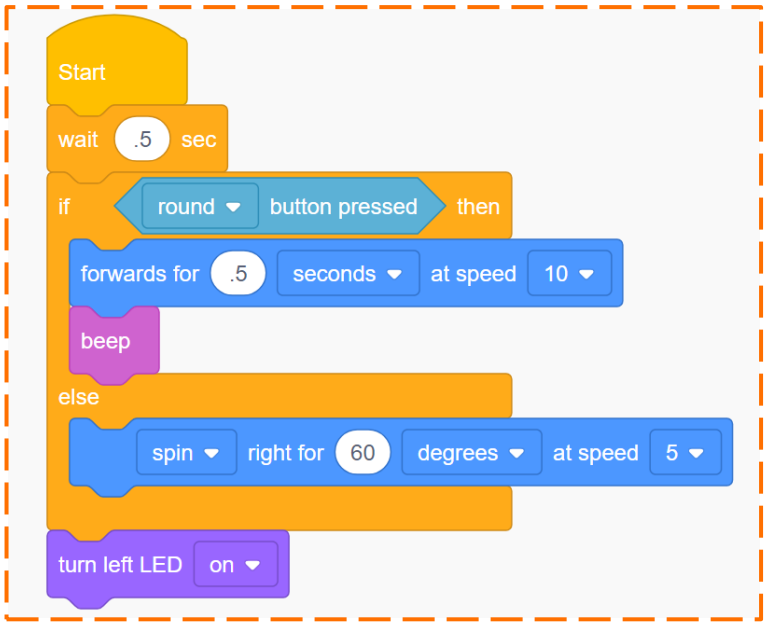
Wenn die Bedingung erfüllt ist, wird der Roboter den Code innerhalb des 'wenn'-Teils des Blocks ausführen. Wenn die Bedingung nicht erfüllt ist, wird der Roboter den Code innerhalb des 'else'-Teils des Blocks ausführen:

Name_____



Die Verwendung einer 'if-else'-Anweisung zwingt ein Programm zum Verzweigen, weil der Roboter nur den 'if'- oder den 'else'-Code ausführen kann, aber nicht beide. Sobald der Roboter entweder den 'if'- oder den 'else'-Code beendet hat, geht er zur nächsten Codezeile im Programm über.

Schaut euch dieses EdScratch-Programm an:



4. Wenn du dieses Programm in Edison ausgeführt hast und der Roboter NICHT einen runden Knopfdruck entdeckt hat, würde der Roboter dann piepen? Warum oder warum nicht?

_____

_____

_____

_____

Name_____

5. Wenn dieses Programm läuft, welche Aktionen werden immer passieren, egal, ob der Roboter einen runden Knopfdruck erkennt oder nicht? Tipp: Folge dem Programm in der Reihenfolge, in der es abläuft. Welche drei Dinge passieren immer, egal was passiert?

_____

_____

_____

Name_____

U4-1.4: Stapeln und Verschachteln von if-Anweisungen

Conditionals sind mächtige Code-Elemente in jeder Computersprache, einschließlich EdScratch. Durch die Verwendung von Conditionals wie 'if'-Anweisungen in EdScratch kannst du alle Arten von interessanten Programmen für deinen Edison-Roboter schreiben.

Alle bedingten Blöcke, einschließlich des 'if'-Blocks und des 'if-else'-Blocks, befinden sich in der Kategorie Control in EdScratch. Schleifen befinden sich ebenfalls in der Kategorie Control. Das liegt daran, dass sowohl die bedingten Blöcke als auch die Schleifen es dir ermöglichen, den Ablauf deines Programms zu kontrollieren. Der if-Block und der if-else-Block haben noch etwas anderes mit Schleifen gemeinsam: Du kannst sie in Programmen stapeln oder verschachteln.

Woran liegt das?

In blockbasierten Programmiersprachen wie EdScratch wird das Zusammenfügen von Blöcken manchmal "Stacking Blocks" genannt. Wenn du mehrere Schleifen in einem Programm nacheinander verwendest, kannst du sagen, dass du die Schleifen stapelst. Du kannst auch if und if-else Blöcke miteinander und mit Schleifen stapeln.

Genauso wie du Schleifen verschachteln kannst, indem du einen Schleifenblock in einen anderen Schleifenblock hineinsteckst, kannst du auch Wenn-Selbst-Blöcke miteinander und mit Schleifen verschachteln!

Name_____

Aufgabe 1: Was wird dieses Mal passieren?

Wenn ein Programm mehrere Schleifen oder bedingte Blöcke gestapelt oder ineinander verschachtelt hat, kann es etwas verwirrend sein, dem Ablauf des Programms zu folgen. Um zu verstehen, was das Programm tun wird, musst du über jede Aktion nachdenken, die nacheinander stattfinden wird.

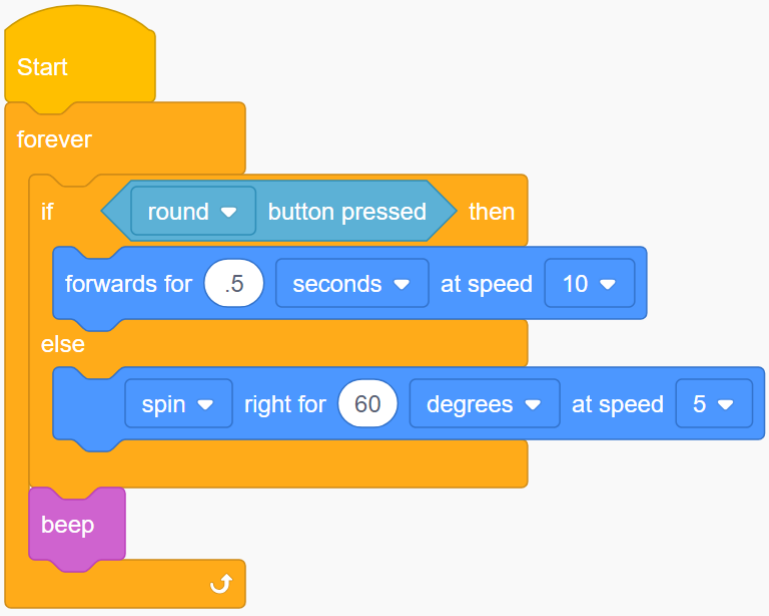
Nicht vergessen

Alle EdScratch Programme, die du machst, funktionieren auf die gleiche grundlegende Weise. Das Programm weist den Roboter an, mit dem obersten Block zu beginnen und dann jede Aktion nacheinander auszuführen. Sobald ein Block ausgeführt wurde, geht das Programm zum nächsten Block über.

Wenn ein Block bedingten Code enthält, prüft das Programm zuerst, ob diese Bedingung erfüllt ist. Das Ergebnis der Bedingungsprüfung bestimmt, welche Aktionen das Programm als nächstes ausführt.

Während Schleifen und Bedingungsblöcke den Ablauf eines Programms kontrollieren, folgen alle Programme immer noch der Reihenfolge nach. Wenn du dir ein Programm mit verschachtelten Schleifen und Bedingungen ansiehst, bedenke, dass dieser Schritt-für-Schritt-Fluss immer stattfindet. Wenn du dich daran erinnerst, wirst du in der Lage sein, zu verfolgen, was in einem Programm vor sich geht. Schau dir die folgenden Programme an und beantworte die Fragen.

Program 1:



1. Wenn du Programm 1 ausführst, aber nie den runden Knopf drückst, was passiert dann? Was passiert dann?

_____

_____

_____

_____

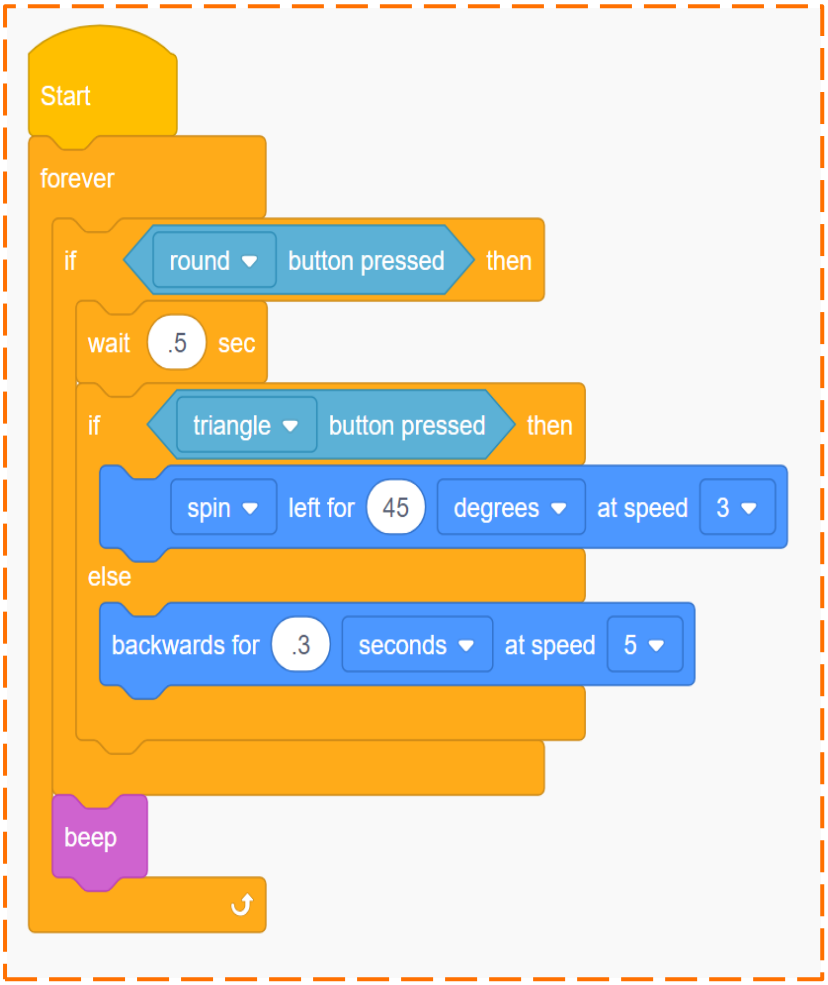
_____

_____

_____

Name_____

Program 2:



2. Wenn du Programm 2 ausführst, aber nie den runden Knopf drückst, was passiert dann? Was passiert dann?

_____

_____

_____

_____

_____

_____

_____

3. Wenn du Programm 2 ausführst, was musst du tun, um den Roboter dazu zu bringen, rückwärts zu fahren? Warum?

_____

_____

_____

_____

_____

_____

_____

_____

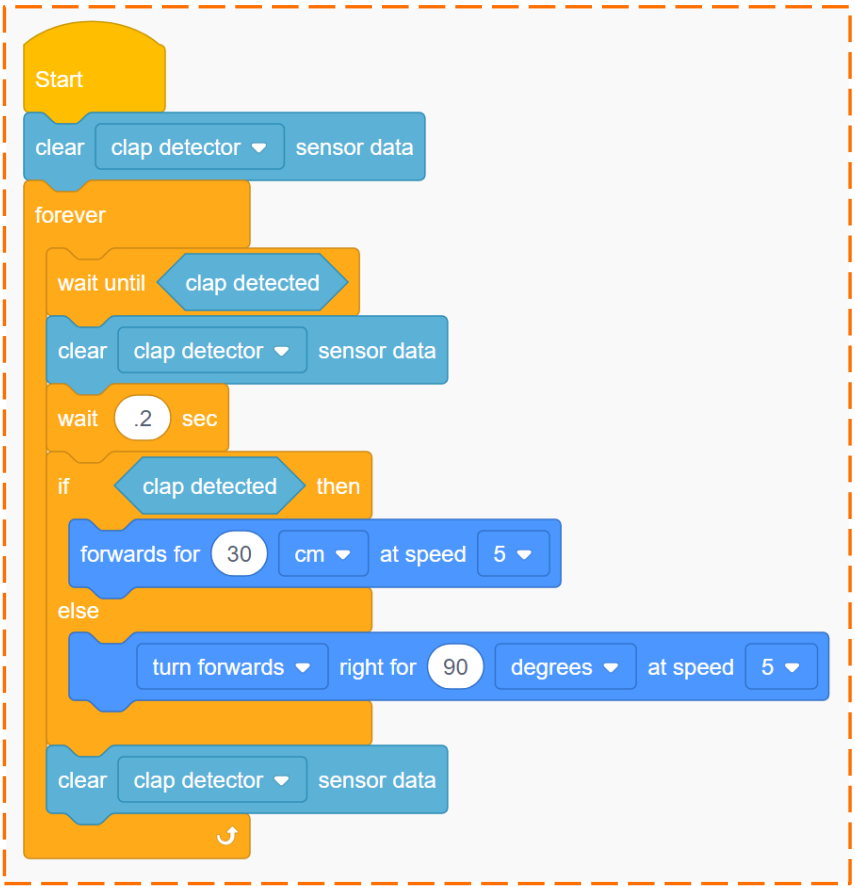
Wähle eines der beiden Programme aus, um in EdScratch zu schreiben. Lade das Programm herunter und teste es in deinem Edison-Roboter. Experimentiere, um zu sehen, wie diese Programme mit verschachtelten 'wenn'-Anweisungen funktionieren.

Name_____

Aufgabe 2: Klatschgesteuertes Fahren

Indem wir einen Wenn-Ansonsten-Block in eine Ewigkeitsschleife (forever) verschachteln, können wir ein klatschgesteuertes Fahrprogramm für Edison bauen. Das Programm muss Edison warten lassen, bis ein Tripper erkannt wird, und dann entweder vorwärts fahren oder sich um 90 Grad drehen, je nachdem, ob der Roboter einen Tripper oder zwei Tripper erkennt.

Schaut euch dieses klatschgesteuerte Fahrprogramm an:



Name_____

Dieses Programm verwendet einen speziellen Block aus der Sensing-Kategorie, den sogenannten Clear Sensor Data Block.

Woran liegt das?

Vergiss nicht, dass dein Edison-Roboter verschiedene Sensoren hat, einschließlich der Technik, die den Roboter Geräusche wie Beifall erkennen lässt. Diese Sensoren erzeugen Daten, wenn sie bestimmte Ereignisse erkennen. Einige dieser Sensordaten werden im Speicher von Edison gespeichert. Diese gespeicherten Daten können manchmal ein Problem darstellen und den Roboter dazu bringen, auf ein altes Ereignis zu reagieren, weil der Roboter sich immer noch an das alte Ereignis 'erinnert'.

Wenn Edison überprüft, ob eine Bedingung erfüllt ist, ob es gespeicherte Daten gibt, wird der Roboter denken, dass die Bedingung erfüllt ist, auch wenn sie nicht erfüllt ist! Deshalb ist es eine gute Codierungspraxis, die Sensordaten zu löschen. Das ist besonders wichtig, wenn man Sensorereignisse in Bedingungen verwendet, die in Schleifen verschachtelt sind. Ihr wollt nicht, dass die Daten aus einer vorherigen Schleife die nächste Schleife beeinflussen!

Es ist auch am besten, die Daten am Anfang eines Programms zu löschen, nur für den Fall, dass der Roboter alte Daten aus einem früheren Programm gespeichert hat.

Sehen Sie sich das klatschgesteuerte Fahrprogramm noch einmal an. Kannst du verfolgen, wie der Code fließt?

Schreibe das klatschgesteuerte Fahrprogramm in EdScratch.

Tipp!

Vergesst nicht, Kommentare zu benutzen! Das Hinzufügen von guten Kommentaren macht es viel einfacher, den Überblick darüber zu behalten, was in einem Programm passieren soll. Dies ist besonders hilfreich in Programmen mit verschachtelten Schleifen und Bedingungen!

Name_____

Lade das Programm auf deinen Edison-Roboter herunter und führe es aus. Experimentiere, um zu sehen, wie das Programm Edison auf Tripper reagieren lässt.

4. Das klatschgesteuerte Fahrprogramm nistet einen Wenn-Ansonsten-Block in eine ewige Schleife ein. Warum glaubst du, dass dies der Fall ist? Was würde passieren, wenn du den Forever-Block nicht benutzen würdest?

_____

_____

_____

_____

_____

_____

Name_____

U4-1.4a: Baue einen Flaschenzug

Eine Umlenkrolle ist eine Vorrichtung, die aus einem Rad mit einer gerillten Felge besteht, über die ein Seil oder eine Kette gezogen wird, um schwere Gegenstände zu heben. Du kannst Edison benutzen, um eine programmierbare Riemenscheibe zu erstellen und ein Programm in EdScratch schreiben, um zu kontrollieren, wie die Riemenscheibe funktioniert!

Was zu tun ist

Baue ein Flaschenzugsystem mit einem Edison-Roboter. Du kannst EdCreate Teile oder jedes andere Material benutzen, das dir gefällt. Du musst auch ein Programm erstellen, um die Riemenscheibe mit EdScratch zu bedienen.

Dein Flaschenzugsystem sollte die Motorausgänge von Edison benutzen, um den Flaschenzug zu steuern. Der Motor sollte die Riemenscheibe in eine Richtung (nach oben oder unten) bewegen, wenn der runde Knopf gedrückt wird und in die andere Richtung, wenn der dreieckige Knopf gedrückt wird.

Tipp!

Versuche, 'if'-Anweisungen in deinem Flaschenzugprogramm zu stapeln oder zu verschachteln, damit der Flaschenzug auf die verschiedenen Knopfdrücke reagiert.

Schreibe dein Programm in EdScratch und teste es mit deinem Riemenscheiben-Design.

Es kann sein, dass du dein Design, dein Programm oder beides ändern musst, damit dein Flaschenzug funktioniert. Das ist in Ordnung! Experimentiere, um zu sehen, was funktioniert.

1. Wie ist es gelaufen? Schreibe über das, was in deinem Flaschenzug -Projekt passiert ist. Was ist schief gelaufen? Wie hast du die Probleme überwunden, auf die du gestoßen bist? Was war der beste Teil des Projekts? Warum war es der beste Teil?

_____

_____

_____

_____

_____

_____

_____

_____

Name_____

U4-2.1: Der Pseudocode

Um gute Computerprogramme zu machen, braucht es mehr als nur das Schreiben von Code. Man muss auch in der Lage sein, Probleme zu lösen, wenn mit einem Programm etwas schief läuft. Deine Programme zu planen, bevor du mit dem Programmieren beginnst, ist eine weitere wichtige und nützliche Fähigkeit beim Programmieren.

Genauso wie man eine Geschichte mit Hilfe eines Storyboards oder einen Aufsatz mit Hilfe einer Gliederung planen kann, ist Pseudocode ein Werkzeug, das Programmierer benutzen, um ihre Programme zu planen, bevor sie mit dem Programmieren beginnen.

Fachjargon

Pseudocode ist eine Möglichkeit, ein Programm in einem einfachen, leicht lesbaren Format zu schreiben. Anstatt sich um die Syntax zu kümmern, verwendet Pseudocode normale Wörter, um zu beschreiben, was das Programm tun wird.
Pseudocode sieht ein bisschen aus wie eine vereinfachte Programmiersprache, basiert aber nicht auf einer bestimmten Programmiersprache. Deshalb kann Pseudocode verwendet werden, um Programme in jeder Programmiersprache zu planen.

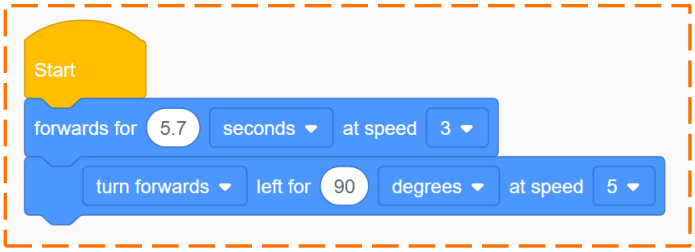
Wenn du dein Programm mit Pseudocode planst, musst du nicht alles im Detail aufschreiben oder dir Gedanken darüber machen, wie es am Ende aussehen wird, wenn du es codierst. Du brauchst nur eine einfache Version deines Plans zu schreiben, die es dir leicht macht, dem Ablauf des Programms zu folgen.

Hier ist ein Beispiel für einen Pseudocode für ein Fahrprogramm für Edison:

```
vorwärts fahren
links abbiegen
```

Der Pseudocode umreißt den grundlegenden Plan, aber er enthält nicht alle Details. Diese werden später ausgearbeitet, wenn du das Programm kodierst.

Hier ist das Programm, in das der Pseudocode übersetzt wurde:



Name_____

Siehst du, wie der Grundplan aus dem Pseudocode in das Programm übersetzt wurde?

Wenn du zuerst einen Plan in Pseudocode schreibst, ist es einfach, die Struktur deines Programms auszuarbeiten. Du kannst ihn dann anpassen und die Details ausfüllen, wenn du den Code schreibst.

Um deinen Pseudocode leicht lesbar zu machen, solltest du ihn ordentlich halten. Schreibe deinen Pseudocode so, dass er genauso fließt wie der Code, wenn du ihn in EdScratch programmierst. Das heißt, du solltest jeden Schritt nacheinander schreiben, Zeile für Zeile.

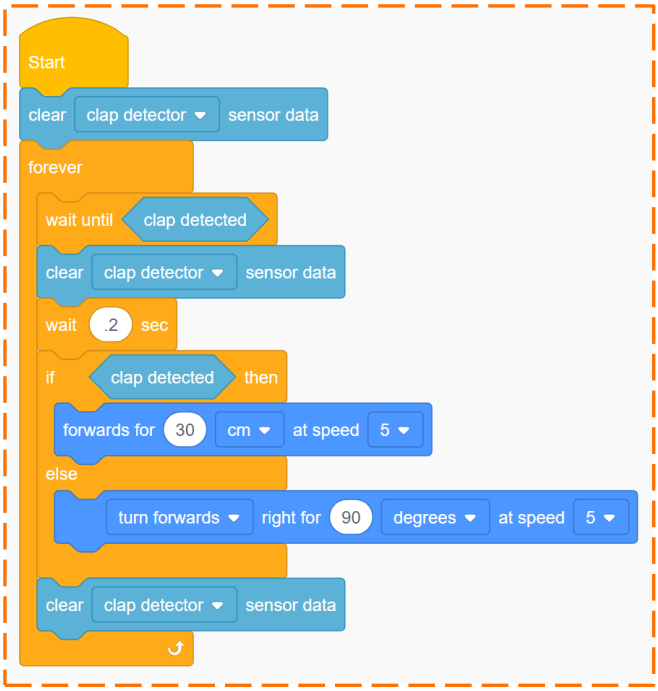
Die Verwendung von Pseudocode ist besonders hilfreich, um Programme zu planen, die Kontrollstrukturen wie Schleifen oder Conditionals verwenden. Du solltest Aktionen einrücken, die sich innerhalb von Schleifen oder Bedingungen befinden, um zu zeigen, dass sich dieser Code innerhalb der Schleife oder der 'if'-Anweisung befindet. Wenn du deinen Pseudocode auf diese Weise organisierst, ist er viel leichter zu verstehen.

Hier ist ein Beispiel für einen Pseudocode, der ein klatschgesteuertes Fahrprogramm und das entsprechende Programm in EdScratch beschreibt:

Pseudocode:

```
clear clap data
forever
        wait until clap
        clear clap data
        wait
        if clap
                drive forward
        else
                turn right
        clear clap data
```

EdScratch-Programm:



Siehst du, wie sich die Aktionen sowohl im Pseudocode als auch im Programm aneinanderreihen?

Name_____

Probier es aus!

Versuchen wir, ein paar Pseudocode-Anweisungen zu lesen. Benutze das Arbeitsblatt U4-2 und folge den Pseudocode-Anweisungen, um die Antworten auf die Fragen zu finden.

Nicht vergessen

Pseudocode sollte immer den grundlegenden Plan umreißen, muss aber nicht alle Details enthalten. Wie viele Details du einfügst, bleibt dir überlassen und kann je nach Programm variieren.

1. Folge dem Pseudocode. Wo wird das Programm enden?

   _____

```
Start on C facing east
forwards until food
left 90 degrees
repeat 6 times
        forwards 1
        if animal
                left 90 degrees
```

2. Folge dem Pseudocode. Wo wird das Programm enden?

   _____

```
Start on H facing east
repeat 3 times
        forwards 1
        if living thing
                right 90 degrees
        else
                left 90 degrees
backwards 1
```

Name_____

3. Folge dem Pseudocode. Wo wird das Programm enden?

   _____

```
Start on D facing west
repeat 3 times
        forwards 3
        if animal
                left 90 degrees
        else
                if food
                        right 180 degrees
repeat 3 times
        forward until letter
        right 90 degrees
backwards until number
```

Tipp!

Denke darüber nach, wie until, if und if-else bedingter Code funktioniert.

Wird das Programm den bedingten Code in einem if-Block ausführen, wenn die Bedingung nicht erfüllt ist? Was passiert stattdessen? Was passiert stattdessen in einem „if-else"-Block?

Stelle sicher, dass du den Pseudocode genau wie ein Computer befolgst!

Name_____

U4-2.1a: Finde die Antwort

Wenn du zum ersten Mal Pseudocode verwendest, mag es ein bisschen unangenehm erscheinen. Wenn du dich aber erst einmal mit Pseudocode vertraut gemacht hast, wirst du viel Zeit sparen, wenn du deine Programme in Pseudocode planst!

Was zu tun ist

Für diese Aktivität musst du mit einem Partner zusammenarbeiten, um das Schreiben und Befolgen von Pseudocode-Anweisungen zu üben. Schreibe mit Hilfe des Arbeitsblattes U4-2 einige Pseudocode-Anweisungen, die dein Partner befolgen soll. In der ersten Zeile deines Pseudocodes sollte deinem Partner mitgeteilt werden, wo er anfangen soll, einschließlich der Richtung, in die er schauen soll.

Nicht vergessen

Achte darauf, dass dein Pseudocode nett und ordentlich und leicht zu lesen ist. Schreibe einen Schritt nach dem anderen, Zeile für Zeile. Du solltest Aktionen einrücken, die sich innerhalb von Schleifen oder Bedingungen befinden, um zu zeigen, dass sich dieser Code innerhalb der Schleife oder der 'if'-Anweisung befindet.

Dein Pseudocode muss nicht alle Details enthalten, sollte aber den Plan klar umreißen, damit dein Partner ihm folgen kann, um die Antworten zu suchen.

1. Schreibe deinen Pseudocode. Teste ihn unbedingt aus, bevor du dich mit deinem Partner austauschst.

Name_____

Mini-Herausforderung!

Kontrollstrukturen, wie Schleifen und Conditionals, machen Programme mächtiger und das Programmieren macht mehr Spaß! Kannst du mindestens zwei verschiedene Kontrollstrukturen in deine Pseudocode-Anweisungen einbauen?

Tipp!

Schaut euch die Kategorie Kontrolle in EdScratch an, um einige Ideen zu erhalten, welche Kontrollstrukturen verwendet werden sollten.

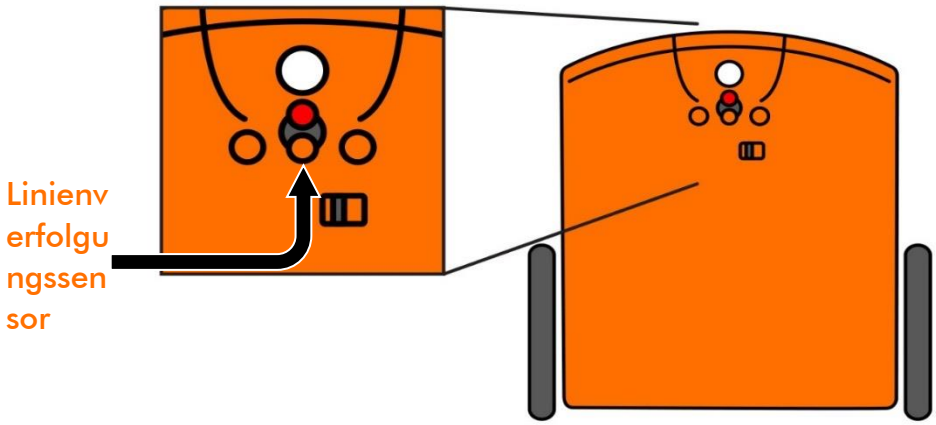
Name_____

U4-2.2: Werfen wir einen Blick auf Edisons Linienverfolger

Edison-Roboter haben verschiedene Sensoren, die verschiedene Dinge erkennen können. Einer dieser Sensoren ist der Linienverfolgungssensor.

Aufgabe 1: Triff Edisons Linienverfolgungssensor

Der Linienverfolgungssensor ist der Sensor, der Edison den Unterschied zwischen dunklen und hellen Oberflächen erkennen lässt. Der Sensor befindet sich auf der Unterseite von



Edison, in der Nähe des Netzschalters.

Der Zeilenverfolgungssensor besteht aus zwei Teilen: einer roten LED und einem Lichtsensor. Schau dir den Linienverfolgungssensor deines Edison-Roboters an. Siehst du die beiden Teile des Sensors?

Der Linienverfolgungssensor funktioniert, indem er Licht von der roten LED auf die Oberfläche unter dem Roboter strahlt. Der Lichtsensor misst dann, wie viel von diesem Licht von der Oberfläche aufprallt. Edison speichert den Wert des reflektierten Lichts als Lichtmesswert. Je mehr Licht zurück zu Edison reflektiert wird, desto höher ist der Lichtwert.

Wird eine weiße oder eine schwarze Oberfläche mehr Licht zu Edison zurückreflektieren? Benutze das Arbeitsblatt U4-3, um zu testen, ob eine weiße oder eine schwarze Oberfläche mehr Licht zu Edison reflektiert.

Schalte Edison ein und drücke zweimal auf den runden Knopf, so dass die rote Linienverfolgungs-LED aufleuchtet. Hebe Edison leicht vom Blatt ab und schaue dir den runden Lichtpunkt, den die LED auf die Oberfläche leuchtet, genau an. Vergleiche, wie hell der Lichtfleck erscheint, wenn er auf eine schwarze und dann auf eine weiße Fläche gelegt wird.

Tipp!

Je mehr Licht reflektiert wird, desto heller wird der Fleck auf der Oberfläche darunter erscheinen.

Name_____

1. Welche Oberfläche reflektiert mehr Licht zurück zu Edison, eine weiße oder eine schwarze Oberfläche? Warum denkst du das?

_____

_____

_____

Indem er misst, wie viel reflektiertes Licht von der Oberfläche unter dem Roboter kommt, lässt der Linienverfolgungssensor den Roboter den Unterschied zwischen dunklen und hellen Oberflächen 'sehen'. Edison sieht jedoch keine Farben wie ein Mensch.

Der Roboter kann nur erkennen, ob eine Oberfläche reflektierend oder nicht reflektierend ist. Eine reflektierende Oberfläche wird viel Licht von der roten LED zurückstrahlen, und eine nicht reflektierende Oberfläche wird sehr wenig Licht zurückstrahlen.

Edison sieht weiße Oberflächen als reflektierend und schwarze Oberflächen als nicht reflektierend. Was ist mit anderen Farben?

2. Wird Edison eine rote Oberfläche als reflektierend oder nicht reflektierend sehen? Wie sieht es mit einer blauen Oberfläche aus? Oder grün? Benutze das Arbeitsblatt U4-3, um alle drei Farben mit der roten Linienverfolgungs-LED zu testen. *Tipp*: Wenn es einen hellen Fleck gibt, der dem ähnelt, was du auf einer weißen Oberfläche siehst, wird viel Licht reflektiert, und der Roboter wird diese Farbe als 'reflektierend' sehen.

Farbe	Reflektierend oder nicht reflektierend?
Rot	
Blau	
Grün	

Name_____

Aufgabe 2: Fahre bis zu einer schwarzen Linie

Wir können Edisons Sensoren benutzen, um Eingaben in EdScratch-Programme zu erstellen, die Edison anweisen, nach verschiedenen Arten von Ereignissen zu suchen und den Roboter anzuweisen, was er tun soll, wenn diese Ereignisse eintreten.

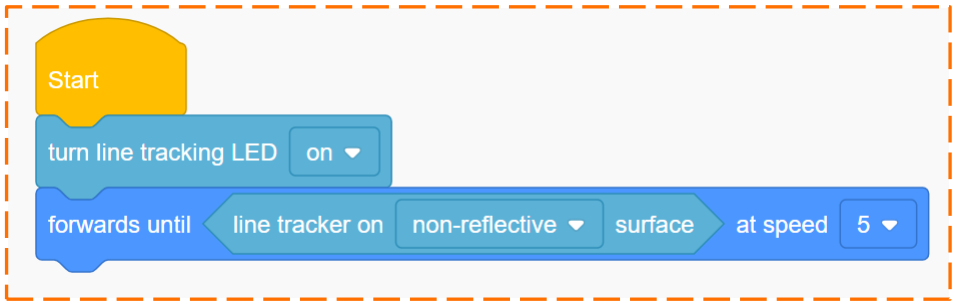
Nicht vergessen

Eingaben sind die Informationen und Anweisungen, die du einem Computer gibst. Wenn du ein Programm für deinen Edison-Roboter schreibst, sagst du dem Roboter, was er tun soll, indem du ihm Eingaben gibst. Edisons Mikrochip verarbeitet dann die Informationen, um dem Roboter zu sagen, was er als Teil des Input-Prozess-Output-Zyklus ausgeben soll.

Ein Ereignis ist etwas, das außerhalb des Programmcodes passiert und das beeinflusst, wie das Programm läuft. Ein Ereignis kann z.B. das Drücken eines Knopfes oder die Übertragung von Informationen von einem Sensor sein.

Versuchen wir, den Linienverfolgungssensor in einem Programm zu verwenden, das Edison anweist, so lange zu fahren, bis es eine schwarze Linie entdeckt. Um dieses Programm zu schreiben, musst du Blöcke aus der Sensing-Kategorie von EdScratch verwenden.

Schau dir dieses Programm an:



Der erste Code-Block in diesem Programm schaltet die Linienverfolgungs-LED an. Wann immer du den Zeilenverfolgungssensor in einem Programm verwenden möchtest, musst du ihn einschalten.

Name_____

Woran liegt das?

Einige von Edisons Sensoren sind immer eingeschaltet und prüfen auf Ereignisse. Der Schallsensor, der Beifälle erkennen kann, ist ein Beispiel für diesen 'immer an' Sensortyp.

Andere Sensoren, wie Edison's Line Tracker, sind standardmäßig ausgeschaltet. Ihr müsst einen Code in euer Programm einbinden, um diese Sensoren einzuschalten. Es reicht jedoch nicht aus, nur die Linetracker-LED einzuschalten. Du brauchst auch einen Code, um dem Sensor mitzuteilen, auf welches Ereignis er achten soll (reflektierende oder nicht reflektierende Oberfläche) und was er tun soll, wenn dieses Ereignis erkannt wird.

Schreibe das Programm in EdScratch und benutze das Arbeitsblatt U4-3, um es mit deinem Edison-Roboter zu testen. Richte deinen Roboter auf den Umriss gegenüber der schwarzen Linie auf dem Arbeitsblatt aus und starte dein Programm. Hält Edison an der schwarzen Linie an?

Mini-Herausforderung!

Wird dein Programm dafür sorgen, dass Edison sowohl an den farbigen Linien auf dem Arbeitsblatt U4-3 als auch an der schwarzen Linie anhält? Warum oder warum nicht?

Überlege dir, ob das Programm Edison bei jeder der Farben stoppen lässt, dann teste, ob du richtig vorausgesagt hast!

Name_____

U4-2.2a: Innerhalb einer Grenze fahren

Du kannst Edisons Linienverfolgungssensor benutzen, um ein Programm zu schreiben, das Edison innerhalb eines schwarzen Rahmens fahren lässt.

Was zu tun ist

Das erste, was du tun musst, ist dein Programm mit Pseudocode zu planen.

Nicht vergessen

Achte darauf, dass dein Pseudocode nett und ordentlich und leicht zu lesen ist. Schreibe einen Schritt nach dem anderen, Zeile für Zeile. Du solltest Aktionen einrücken, die sich innerhalb von Schleifen oder Bedingungen befinden, um zu zeigen, dass sich dieser Code innerhalb der Schleife oder der 'if'-Anweisung befindet.

Dein Pseudocode muss nicht alle Details enthalten, aber er sollte deinen Plan klar umreißen, damit du ihn später zum Schreiben des Codes verwenden kannst.

Dein Programm sollte Edison laufen lassen, bis es eine schwarze Linie erkennt. Wenn der Roboter eine schwarze Linie entdeckt, sollte er zurückfahren, dann von der Linie wegdrehen und dann wieder losfahren, bis er eine schwarze Linie entdeckt.

1. Schreibe deinen Pseudocode.

_____

_____

_____

_____

_____

_____

Benutze deinen Pseudocode als Anleitung, um dein Programm in EdScratch zu schreiben. Lade ihn auf deinen Edison-Roboter herunter und teste ihn mit dem Arbeitsblatt U4-4.

Name_____

Tipp!

Wenn dein Programm beim ersten Mal nicht funktioniert, ist das in Ordnung! Überprüfe die Logik deines Pseudocodes, um zu sehen, ob du irgendwelche Probleme erkennen kannst. Vergiss nicht, auch in der Bug-Box nach Nachrichten zu suchen!

Name_____

U4-2.3: Betrachten wir Algorithmen näher

Edisons Linienverfolgungssensor kann erkennen, ob die Oberfläche unter dem Roboter reflektierend oder nicht reflektierend ist. Du kannst diesen Sensor benutzen, um Edison dazu zu bringen, sich auf verschiedene Weise zu verhalten, z.B. so lange zu fahren, bis er eine schwarze Linie entdeckt, und dann anzuhalten. Du kannst diesen Sensor auch benutzen, um Edison so zu programmieren, dass er einer schwarzen Linie folgt, auch wenn du nicht weißt, wie diese schwarze Linie aussieht. Um dies zu tun, musst du zuerst einen Algorithmus erstellen.

Fachjargon

Ein Algorithmus ist eine breite Palette von Anweisungen, um eine Reihe von Problemen zu lösen. Ein Algorithmus legt einen Prozess oder eine Reihe von Regeln fest, die befolgt werden müssen, um ein Problem aus der Menge zu lösen.

Computerprogramme verwenden oft Algorithmen, aber Programme und Algorithmen sind nicht dasselbe. Denke daran, dass ein Computerprogramm eine Sammlung von Anweisungen ist, die einen Computer anweisen, eine bestimmte Aufgabe auszuführen. Ein Algorithmus legt die Logik dafür fest, wie eine ganze Reihe von Problemen zu lösen ist, nicht nur eine bestimmte Aufgabe. Du kannst ein Computerprogramm schreiben, das einen Algorithmus verwendet, aber nicht alle Computerprogramme sind Algorithmen.

Algorithmen sind wirklich hilfreich, denn mit einem Algorithmus kann eine Person oder ein Computer eine ganze Reihe von Problemen lösen, auch wenn du nicht jedes Detail kennst.

Name_____

Woran liegt das?

Nehmen wir an, du willst deinen Freunden beibringen, wie man Obstkuchen macht. Wenn du weißt, dass alle deine Freunde Äpfel haben, kannst du einfach ein Rezept für Apfelkuchen aufschreiben.

Vielleicht haben aber nicht alle deine Freunde Äpfel. Was ist, wenn einer deiner Freunde Blaubeeren hat, ein anderer Kirschen und ein dritter Äpfel? Sie können nicht alle das Rezept für Apfelkuchen befolgen. Du müsstest für jede Frucht ein eigenes Rezept schreiben.

Was, wenn du nicht weißt, welche Frucht jeder deiner Freunde hat? Wie könntest du ihnen beibringen, Obstkuchen zu machen?

Egal, welche Früchte sie haben, alle deine Freunde müssen die gleichen grundlegenden Anweisungen befolgen: Teig herstellen, Kuchen mit den Früchten füllen und dann backen.

Diese neue Anleitung ist ein Beispiel für einen Algorithmus.

In der Computerprogrammierung wollen wir oft Anweisungen für einen Computer erstellen, die dieser befolgen soll, um eine ganze Reihe von Problemen zu lösen. Indem wir einen Algorithmus verwenden, können wir ein Programm schreiben, mit dem der Computer jedes Problem in der Menge lösen kann. Ohne einen Algorithmus müssten wir für jedes Problem einzeln ein neues Programm schreiben.

Name_____

Aufgabe 1: Folge einer schwarzen Linie

Du weißt, dass du Edisons Linienverfolgungssensor benutzen kannst, um schwarze (nicht reflektierende) und weiße (reflektierende) Oberflächen zu erkennen. Wir können einen Algorithmus erstellen, der diesen Sensor benutzt, um Edison dazu zu bringen, jeder schwarzen Linie zu folgen.

Woran liegt das?

Sagen wir, du zeichnest eine schwarze Linie, der Edison folgen soll. Um Edison dazu zu bringen, deiner Linie zu folgen, könntest du ein Programm schreiben, das Edison dazu bringt, den genauen Weg der Linie zu fahren. Wenn du jedoch eine neue Linie zeichnest, musst du ein ganz neues Programm für diese neue Linie schreiben.

Stattdessen kannst du einen Algorithmus erstellen.

Dieser Algorithmus wird eine Reihe von Problemen lösen: 'folge jeder schwarzen Linie'. Jede spezifische Linie, die du für Edison machst, um ihr zu folgen, ist ein neues Problem in diesem Satz.

Wenn du den Algorithmus benutzt, um die Logik zu steuern, kannst du dann ein Programm schreiben, das für alle Probleme in diesem Set funktioniert. Auf diese Weise wird nicht für jedes neue Problem ein ganz neues Programm benötigt.

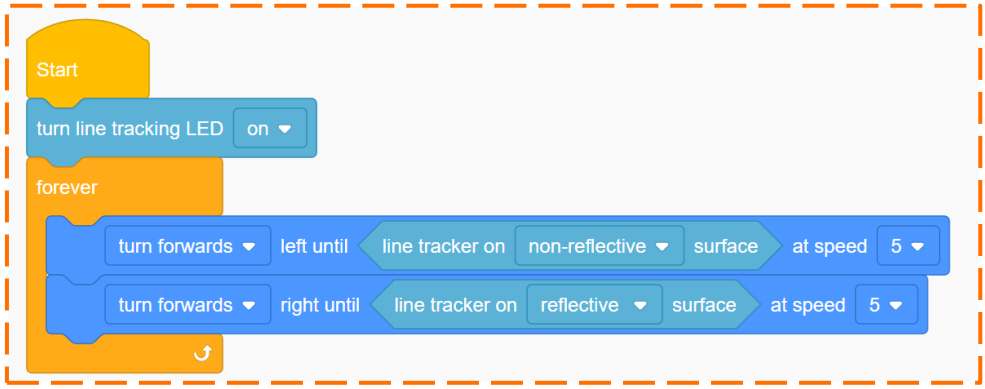
Du kannst einen Algorithmus mit Hilfe von Pseudocode planen, genau wie du es tust, wenn du ein Programm planst. Hier ist ein Algorithmus in Pseudocode, der Edison erlaubt, jeder schwarzen Linie zu folgen:

```
Line Tracker einschalten
Schleife für immer
        Fahre vorwärts nach links, bis der Roboter eine schwarze Oberfläche
entdeckt.
        Fahre nach rechts vorwärts, bis der Roboter eine weiße Oberfläche
entdeckt.
```

Dieser Algorithmus besagt, dass der Roboter vorwärts nach links fahren soll, bis sich der Linienverfolgungssensor auf einer nicht reflektierenden (schwarzen) Oberfläche befindet. Dann sollte der Roboter vorwärts nach rechts fahren, bis der Line Tracking Sensor auf

einer reflektierenden (weißen) Oberfläche ist. Der Roboter sollte dieses Verhalten für immer beibehalten.

Dieser Algorithmus kann verwendet werden, um ein Programm in EdScratch zu schreiben:



Kannst du sehen, wie die Logik des Algorithmus innerhalb des Programms ist?

Schreibe das zeilenverfolgende Programm in EdScratch und lade es auf deinen Edison-Roboter herunter. Benutze das Arbeitsblatt U4-4, um das Programm zu testen. Denke daran, Edison mit dem Linienverfolger auf einer weißen Fläche zu starten, nicht direkt auf der schwarzen Linie.

1. Wie bewegt sich der Roboter, wenn du das Programm ausführst? Schau dir das Programm an und denke über die Logik im Algorithmus nach. Warum bewegt sich der Roboter auf diese Weise?

_____

_____

_____

_____

Name_____

Aufgabe 2: Folge einer anderen schwarzen Linie

Das Programm, das du geschrieben hast, benutzt einen Algorithmus, der dazu bestimmt ist, jedes Problem aus der Menge der Probleme zu lösen: 'folge jeder schwarzen Linie'. Das bedeutet, dass dasselbe Programm Edison jeder schwarzen Linie folgen lassen sollte, die du machst!

Erstelle deine eigene Linie zum Testen. Benutze einen schwarzen Marker auf weißem Papier oder mache eine Linie mit schwarzem Klebeband auf dem Boden oder einem Schreibtisch. Lass das Programm in Edison laufen, um deine Linie zu testen. Kann Edison deiner Linie folgen?

2. War Edison in der Lage, deiner Linie zu folgen? Wenn du irgendwelche Probleme hattest, beschreibe sie. Was denkst du, was die Probleme verursacht hat?

_____

_____

_____

Name_____

U4-2.3a: Es gibt mehr als einen Weg, einer Linie zu folgen

Um deinen Edison-Roboter dazu zu bringen, einer beliebigen Linie zu folgen, musst du einen Algorithmus verwenden.

Nicht vergessen

Ein Algorithmus ist eine breite Palette von Anweisungen, um eine Reihe von Problemen zu lösen. Ein Algorithmus legt die Logik dafür fest, wie eine ganze Reihe von Problemen zu lösen ist, nicht nur eine bestimmte Aufgabe.

Denke über die Logik im Algorithmus nach, die es Edison erlaubt, jeder schwarzen Linie zu folgen. Im einfachsten Fall, was sagt er aus?

Die groben Anweisungen zur Lösung des Problems 'folge jeder schwarzen Linie' sind: Benutze den Line Tracker, während du dich vorwärts bewegst, gehe in eine Richtung, wenn du auf schwarzem Weg bist und in die andere Richtung, wenn du auf weißem Weg bist. Hier ist, wie dieser sehr einfache Algorithmus in Pseudocode aussieht:

```
Benutze den Line Tracker
für immer
    wenn der Roboter Schwarz erkennt, fahre in eine Richtung vorwärts
    (links oder rechts)
    wenn der Roboter Weiß erkennt, fahre in die andere Richtung (rechts
    oder links)
```

Wenn du einen Algorithmus planst, um ihn dann zu programmieren, könntest du dieselbe Logik etwas anders schreiben, und du würdest wahrscheinlich einige der Details hinzufügen, wie in diesem Beispiel:

```
Line Tracker einschalten
Schleife für immer
    Fahre vorwärts nach links, bis der Roboter eine schwarze
    Oberfläche entdeckt.
    Fahre nach rechts vorwärts, bis der Roboter eine weiße
    Oberfläche entdeckt.
```

Da die Logik in beiden Beispielen die gleiche ist, werden alle Programme, die du mit beiden Beispielen erstellst, der gleichen Logik folgen. Selbst wenn die Programme

unterschiedlich aussehen, werden sie immer noch die 'folge jeder schwarzen Linie' Probleme lösen.

Wie viele verschiedene Möglichkeiten gibt es, ein Programm zu schreiben, das die Probleme der 'folge jeder schwarzen Linie' löst? Mehr als du vielleicht denkst!

Was zu tun ist

Deine Herausforderung besteht darin, mindestens zwei verschiedene Programme zu schreiben, die die grundlegende Logik des 'folge jeder schwarzen Linie'-Algorithmus verwenden.

Du musst jedes Programm mit Pseudocode planen und es dann in EdScratch programmieren.

Tipp!

Pseudocode hilft uns bei der Planung, Kommentare helfen uns beim Debuggen. Das Hinzufügen von Kommentaren während des Programmierens hilft dir, deine Logik zu überprüfen und dein Denken zu verfolgen, damit du alle Probleme, auf die du beim Testen deines Programms stößt, suchen und beheben kannst.

Lade jedes deiner Programme einzeln herunter. Teste jedes Programm mit dem Arbeitsblatt U4-4 oder erstelle deine eigene Zeile, die du als Testraum verwenden kannst. Deine beiden Programme müssen Edison's Zeilenverfolgungssensor verwenden und sollten in der Lage sein, jeder schwarzen Linie zu folgen.

Tipp!

Denke darüber nach, wie du die Logik des Algorithmus in EdScratch anwenden kannst. Betrachte verschiedene Bedingungen, einschließlich until blocks, if blocks und if-else blocks. Vergiss auch die Kategorie Ereignisse nicht!

Name_____

Wie sehen deine beiden Programme aus? Entweder schreibst du Pseudocode, lädst deine Programmdateien herunter und teilst sie mit anderen oder du schreibst eine Erklärung für jedes deiner Programme.

Program 1:

Program 2:

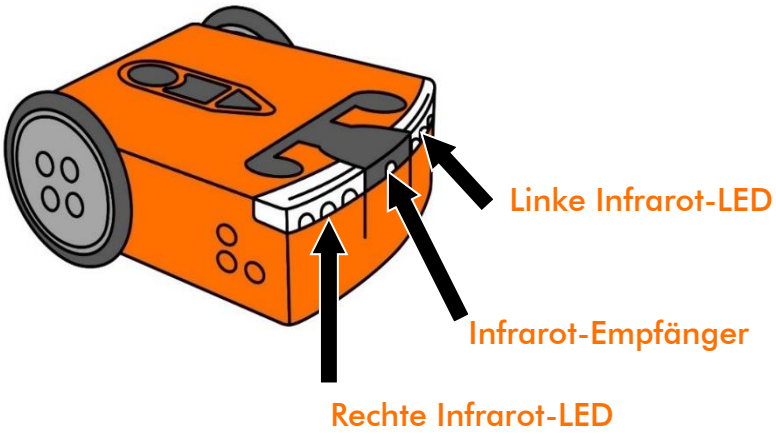
Name_____

U4-2.4: Edisons Hinderniserkennung

Edison-Roboter haben verschiedene Sensoren, die verschiedene Dinge erkennen können. Einer dieser Sensoren ist der Infrarotlichtsensor, mit dem wir Hindernisse erkennen können.

Aufgabe 1: Edisons Infrarot-Lichtsensor

Edisons Infrarot-Lichtsensor ist der Sensor, der Edison Infrarotlicht ausstrahlen und erkennen lässt. Der Sensor besteht aus drei Teilen, die sich alle auf der Vorderseite von Edison befinden: zwei Infrarot-LEDs (eine rechts und eine links) und ein Infrarotempfänger in der Mitte. Schau dir deinen Edison-Roboter an. Siehst du die verschiedenen Teile des Sensors?



Genau wie die beiden roten LEDs von Edison können die beiden Infrarot-LEDs Licht aussenden. Weil es aber infrarot ist, wirst du es nicht sehen können.

Woran liegt das?

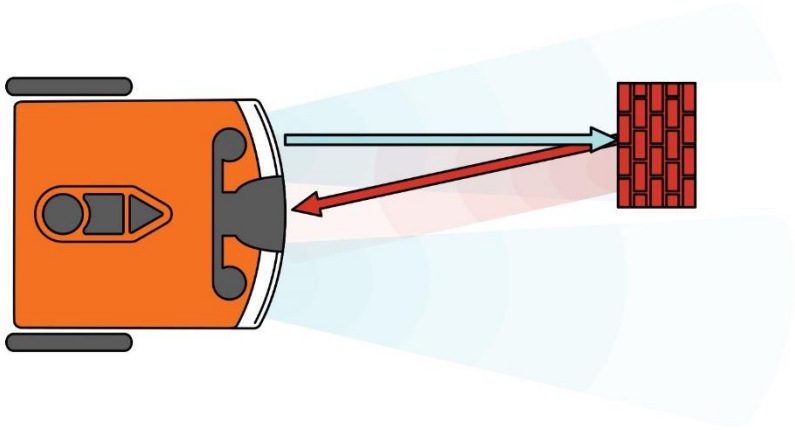
Es gibt eine breite Palette von Licht. Die Menschen können einen Teil dieser Bandbreite sehen, aber nicht alles. Infrarotes Licht (das auch IR-Licht genannt wird) ist für Menschen nicht sichtbar.

Auch wenn wir es nicht sehen können, benutzen die Menschen Infrarot sehr viel. Zum Beispiel wird Infrarotlicht in Fernbedienungen von Fernsehern verwendet. Auf diese Weise sagt die Fernbedienung dem Fernseher, dass er den Kanal wechseln oder die Lautstärke erhöhen soll!

Name_____

Im Gegensatz zu uns kann Edison Infrarotlicht mit dem Infrarotempfänger an der Vorderseite des Roboters erkennen. Eine Möglichkeit, wie wir den Infrarotlichtsensor von Edison benutzen können, ist die Erkennung von Hindernissen.

Edison kann Infrarotlicht von den beiden Infrarot-LEDs aussenden. Wenn dieses Infrarotlicht auf ein Hindernis trifft, wie zum Beispiel eine Wand, wird das Licht zurück in Richtung Edison reflektiert. Der Infrarotempfänger von Edison erkennt das reflektierte Licht und teilt Edison mit, dass es ein Hindernis gibt.



Je nachdem, wo sich das Hindernis befindet, wird das Infrarotlicht von der linken LED, der rechten LED oder von beiden zurückgestrahlt. Das reflektierte Licht teilt dem Roboter mit, wo sich das Hindernis befindet.

Aufgabe 2: Vorwärts bis zum Hindernis

Wir können Edisons Sensoren benutzen, um Eingaben in EdScratch-Programme zu erstellen, die Edison anweisen, nach verschiedenen Arten von Ereignissen zu suchen und den Roboter anzuweisen, was er tun soll, wenn diese Ereignisse eintreten.

Nicht vergessen

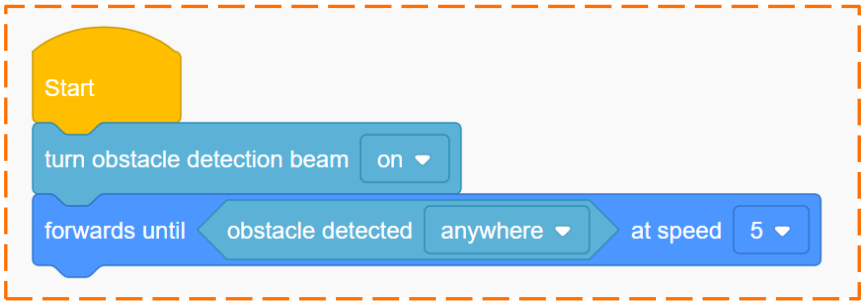
Eingaben sind die Informationen und Anweisungen, die du einem Computer gibst. Wenn du ein Programm für deinen Edison-Roboter schreibst, sagst du dem Roboter, was er tun soll, indem du ihm Eingaben gibst. Edisons Mikrochip verarbeitet dann die Informationen, um dem Roboter zu sagen, was er als Teil des Input-Prozess-Output-Zyklus ausgeben soll.

Ein Ereignis ist etwas, das außerhalb des Programmcodes passiert und das beeinflusst, wie das Programm läuft. Ein Ereignis kann z.B. das Drücken eines Knopfes oder die Übertragung von Informationen von einem Sensor sein.

Name_____

Lass uns versuchen, den Infrarotsensor in einem Programm zu verwenden, das Edison anweist, so lange zu fahren, bis er ein Hindernis erkennt. Um dieses Programm zu schreiben, müsst ihr Blöcke aus der Sensing-Kategorie von EdScratch verwenden.

Schau dir dieses Programm an:



Der erste Codeblock in diesem Programm schaltet den Hinderniserkennungsstrahl ein.

Woran liegt das?

Einige von Edisons Sensoren sind immer eingeschaltet und prüfen auf Ereignisse. Der Infrarotempfänger ist eigentlich immer an, aber die Infrarot-LEDs sind es nicht.

Damit Edison Hindernisse erkennen kann, musst du Code in dein Programm einbinden, um den Strahl zur Hinderniserkennung einzuschalten. Dieser schaltet die Infrarot-LEDs ein und weist den Infrarotempfänger an, nach reflektiertem Infrarotlicht zu suchen, das zum Roboter zurückprallt. Es reicht jedoch nicht aus, nur den Hinderniserkennungsstrahl einzuschalten. Man braucht auch einen Code, der dem Sensor sagt, auf welches Ereignis er achten soll (mit anderen Worten, wo er nach einem Hindernis suchen soll) und was er tun soll, wenn dieses Ereignis erkannt wird.

Name_____

Schreibe das Programm in EdScratch. Lade das Programm herunter und teste es mit deinem Edison-Roboter. Du brauchst auch ein Hindernis, um es in deinem Test zu verwenden.

Einige Hindernisse werden mit Edison besser funktionieren als andere. Wenn ein Hindernis zu klein ist oder nicht genug Infrarotlicht reflektiert, kann Edison es nicht erkennen. Wähle ein Objekt, das undurchsichtig, aber nicht zu dunkel (wähle keine schwarzen Objekte) und mindestens so groß wie Edison ist. Für dieses Programm wäre die Wand des Raumes ein gutes Hindernis.

Teste das Programm, um zu sehen, wie es funktioniert.

Tipp!

Wenn dein Roboter dein Hindernis nicht erkennt, versuche es mit einem anderen Objekt. Wenn er das Hindernis immer noch nicht erkennt, musst du eventuell die Hinderniserkennung kalibrieren. Frag deinen Lehrer nach dem speziellen Strichcode, um die Hinderniserkennung zu kalibrieren.

Aufgabe 3: Erkennen und vermeiden

Anstatt einfach anzuhalten, wenn er ein Hindernis entdeckt, kannst du Edison dazu bringen, jedem Hindernis auszuweichen und weiterzufahren. Um dies zu tun, musst du einen Algorithmus erstellen, der die folgenden Probleme löst: 'erkenne und vermeide jedes Hindernis'.

1. Benutze Pseudocode, um einen Algorithmus zum Erkennen und Ausweichen zu schreiben.

_____

_____

_____

_____

_____

Sobald du deinen Algorithmus hast, schreibe ein EdScratch-Programm für Edison, das die Logik dieses Algorithmus verwendet. Lade dein Programm herunter und teste es mit Edison und einigen Hindernissen.

2. Selbst wenn professionelle Computerprogrammierer Programme schreiben, stoßen sie auf Probleme. Beschreibe ein Problem, das du bei der Erstellung deines Erkennungs- und Vermeidungsalgorithmus oder Programms hattest. Was hast du getan, um das Problem zu lösen?

_____

_____

_____

_____

Name_____

U4-2.4a: Schneller, schneller, smash?

Edisons Hinderniserkennung funktioniert, indem er Infrarotlicht von den beiden Infrarot-LEDs des Roboters aussendet. Wenn es ein Hindernis gibt, prallt das Licht von dem Hindernis ab und reflektiert zurück zu Edison, wo der Infrarotempfänger des Roboters das Hindernis erkennt. Wir können Edison so programmieren, dass er das reflektierte IR-Licht erkennt und auf dieses Ereignis reagiert.

Roboter können Informationen unglaublich schnell verarbeiten, aber die Aufgabe, Eingaben zu empfangen, die Informationen zu verarbeiten und eine Ausgabe zu erzeugen, nimmt immer noch einige Zeit in Anspruch. Es gibt mehrere Schritte im Hinderniserkennungsprozess des Edison-Roboters: Edison sendet IR-Licht aus, das Licht trifft auf ein Objekt, das Licht prallt von dem Objekt ab, der Infrarotempfänger des Roboters erkennt das reflektierte Licht, und schließlich reagiert der Roboter auf dieses Ereignis. Wie lange dauert dieser Prozess der Hinderniserkennung?

Was ist zu tun?

Schreibe ein Programm, um Edison dazu zu bringen, vorwärts zu fahren, bis der Roboter ein Objekt entdeckt. Edison soll aufhören zu fahren, sobald er das Objekt entdeckt hat, damit der Roboter nicht auf das Hindernis trifft.

Lade dein Programm herunter und teste es mit Edison und einem Objekt, das Edison entdecken können sollte. Stelle dann den Parameter für die Geschwindigkeitseingabe in deinem Programm ein. Teste verschiedene Geschwindigkeiten mit Edison, um zu sehen, was bei schnelleren und langsameren Geschwindigkeiten passiert.

Nicht vergessen

Immer wenn ihr den Infrarotsensor zur Hinderniserkennung benutzen wollt, müsst ihr den Hinderniserkennungsstrahl einschalten.

1. Was passiert, wenn du dein Hinderniserkennungsprogramm mit einer sehr hohen Geschwindigkeit ausführst?

_____

_____

2. Beim Programmieren müssen wir manchmal verschiedene Features ausbalancieren, um das beste Ergebnis zu erzielen. Dies ist als Kompromiss bekannt. Beschreibe den Kompromiss zwischen der Geschwindigkeit von Edison und der Fähigkeit des Roboters, Hindernisse zu erkennen.

_____

_____

_____

_____

Name_____

U4-2.4b: Wenn Linie, nach rechts gehen. Wenn Hindernis, gehe links

Edison-Roboter haben verschiedene Sensoren, die verschiedene Dinge erkennen können. Du kannst mehrere Sensoren in einem einzigen Programm verwenden und so den Roboter dazu bringen, auf verschiedene Arten von Ereignissen zu reagieren.

Was zu tun ist

Schreibe ein Programm, so dass sich dein Edison-Roboter durch ein Gitter bewegt und jeden neuen Abschnitt des Gitters auf nicht reflektierende (schwarze) Oberflächen oder Hindernisse überprüft. Dein Programm sollte Edison mitteilen, dass der Roboter nach rechts abbiegen muss, wenn er eine schwarze Oberfläche entdeckt, aber wenn er ein Hindernis entdeckt, sollte er stattdessen nach links abbiegen. Teste dein Programm mit dem Arbeitsblatt U4-5. Versuche, ob du ein Programm schreiben kannst, so dass dein Edison-Roboter auf dem Umriss startet und dann seine Sensoren benutzt, um zum Ziel der Parkzone zu gelangen.

Tipp!

- Sensoren wie der Hinderniserkennungsstrahl und der Line Tracker müssen eingeschaltet sein, damit sie funktionieren. Dein Programm muss Edison auch sagen, auf welches Ereignis die Sensoren achten sollen und wie sie auf dieses Ereignis reagieren sollen.
- Einige der Sensoren von Edison, einschließlich der Hinderniserkennung, erzeugen und speichern Daten, wenn sie bestimmte Ereignisse erkennen. Es ist gute Codierpraxis, die Sensordaten zu löschen, besonders wenn man Sensorereignisse in Bedingungen verwendet, die in Schleifen verschachtelt sind. Du willst nicht, dass die Daten aus einer vorherigen Schleife die nächste Schleife beeinflussen!
- Es ist auch ratsam, die Sensordaten beim Start eines Programms zu löschen, nur für den Fall, dass der Roboter alte Daten aus einem früheren Programm gespeichert hat.
- Pseudocode hilft uns bei der Planung, Kommentare helfen uns beim Debuggen. Benutze sie, um dein Programm zu planen und zu testen!

Name_____

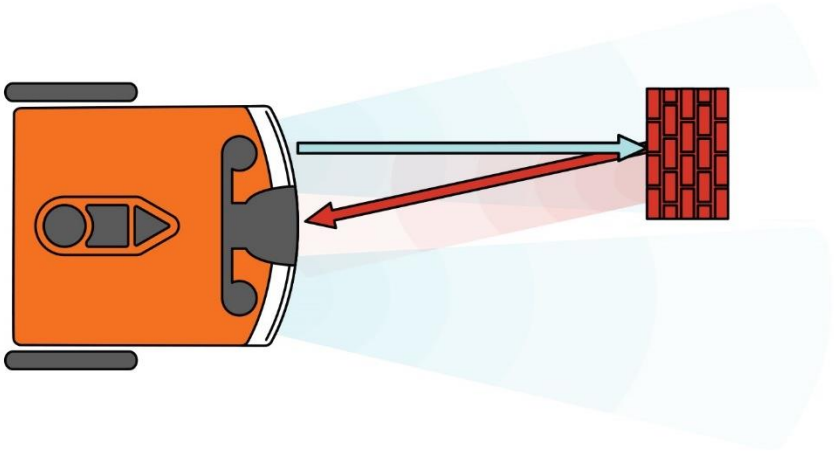
Mini-Herausforderung!

Kannst du ein eigenes 'Gitterlabyrinth' erstellen, das Edison mit seinen Sensoren lösen kann? Entwerfe eine Anlage, die einen Start, ein Ziel und sowohl schwarze Flächen als auch Hindernisse zu erkennen hat. Schreibe ein Programm, das die Sensoren des Roboters benutzt, um Edison zum Tor zu bringen.

Name_____

U4-2.4c: Wo ist das Hindernis?

Edison kann Infrarotlicht von den beiden Infrarot-LEDs rechts und links des Roboters aussenden. Wenn dieses Infrarotlicht auf ein Hindernis trifft, wie eine Wand oder deine Hand, wird das Licht zurück in Richtung Edison reflektiert. Der Infrarotempfänger von Edison erkennt das reflektierte Licht und teilt Edison mit, dass es ein Hindernis gibt.



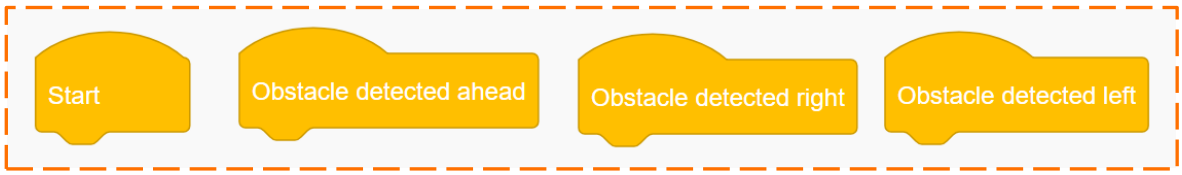
Je nachdem, wo sich das Hindernis befindet, wird das Infrarotlicht von der linken LED, der rechten LED oder von beiden zurückreflektiert. Das reflektierte Licht teilt dem Roboter mit, wo sich das Hindernis befindet.

Du kannst Edison so programmieren, dass er mit verschiedenen Ausgängen reagiert, je nachdem, wo der Roboter das Hindernis erkennt.

Was zu tun ist

Erstelle ein Programm in EdScratch, um Edison dazu zu bringen, Hindernisse zu erkennen, aber reagiere mit einer unterschiedlichen Ausgabe, je nachdem, wo sich das Hindernis befindet: rechts, links oder geradeaus.

In dieser Herausforderung musst du Blöcke aus der Kategorie Events verwenden. Dein Programm sollte alle Blöcke auf diesem Bild enthalten:



Name_____

Was der Roboter für jedes der verschiedenen Ereignisse ausgibt, ist euch überlassen.

Tipp!

Wahrscheinlich willst du Edisons Motoren nicht als Ausgänge in diesem Programm verwenden. Warum nicht? Wenn sich der Roboter bewegt, wird er seine Position relativ zum Hindernis verändern. Das könnte wirklich verwirrend werden!

Welche Ausgänge könntest du stattdessen benutzen? Was könntest du tun, damit deine Ausgänge helfen zu kommunizieren, wo der Roboter das Hindernis entdeckt hat?

Lade dein Programm herunter und teste es, indem du Objekte nach rechts, dann nach links und dann geradeaus vor Edison stellst.

Name_____

U4-2.4d: 3D-Labyrinth

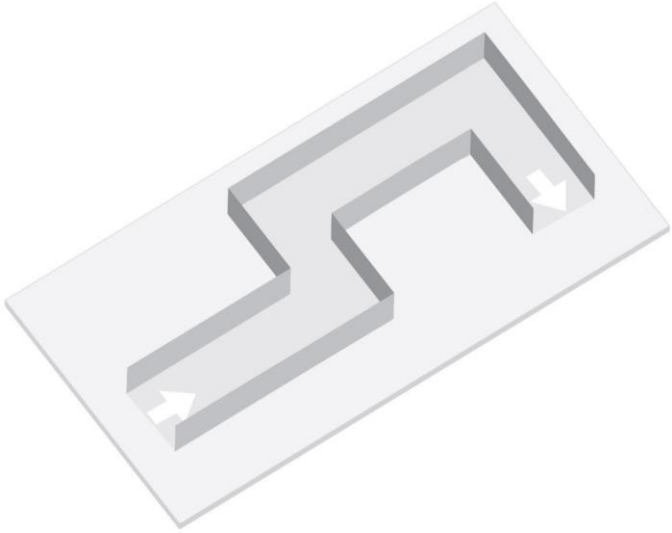
Du kannst Edisons Hinderniserkennungsstrahl benutzen, um den Roboter so zu programmieren, dass er Hindernisse erkennt und darauf reagiert. Kannst du die Hinderniserkennung benutzen, um Edison dazu zu bringen, autonom durch ein Labyrinth zu fahren?

Was ist zu tun?

In dieser Herausforderung müsst ihr ein 3D-Labyrinth bauen, in dem Edison mit Hilfe von Hinderniserkennung navigieren kann. Die Wände deines Labyrinths müssen hoch genug sein, damit Edison sie erkennen und darauf reagieren kann. Außerdem musst du ein Programm schreiben, um Edison mit Hilfe der Hinderniserkennung durch das Labyrinth zu bringen.

Tipp!

- Einige der Sensoren von Edison, einschließlich der Hinderniserkennung, erzeugen und speichern Daten, wenn sie bestimmte Ereignisse erkennen. Es ist gute Codierpraxis, die Sensordaten zu löschen, besonders wenn man Sensorereignisse in Bedingungen verwendet, die in Schleifen verschachtelt sind, sowie beim Start eines Programms.
- Denkt daran, dass Edison erkennen kann, wo sich ein Hindernis befindet: rechts, links oder geradeaus. Dies kann dir helfen, ein erfolgreiches Programm zu erstellen.
- Pseudocode hilft uns bei der Planung, Kommentare helfen uns beim Debuggen. Benutze sie, um dir zu helfen, dein Programm zu planen und zu testen!



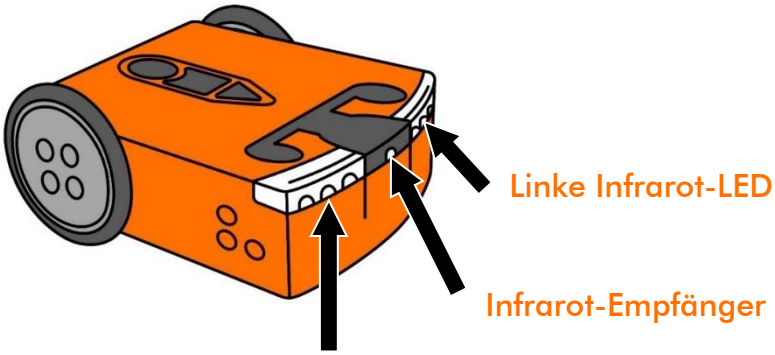
Name_____

U4-2.5: Nachrichtenaustausch mit Edison

Edison-Roboter haben einen Infrarot-Lichtsensor, mit dem Hindernisse erkannt werden können. Wir können diesen Sensor auch auf eine andere Art und Weise nutzen: zum Senden und Empfangen von Infrarot-Nachrichten.

Aufgabe 1: Edison und Infrarot-Nachrichten

Edisons Infrarot (IR)-Lichtsensor ist der Sensor, der Edison Infrarotlicht ausstrahlen und erkennen lässt. Der Sensor besteht aus drei Teilen, die sich alle auf der Vorderseite von Edison befinden: zwei Infrarot-LEDs (eine rechts und eine links) und ein Infrarotempfänger in der Mitte.



Bei der Hinderniserkennung verwenden wir die Infrarot-LEDs, um Infrarotlicht auszusenden, und den Infrarotempfänger, um zu überprüfen, ob dieses Licht von Gegenständen zurück nach Edison reflektiert wurde. Wir können den Infrarotempfänger auch benutzen, um Infrarotlicht von anderen Quellen zu erkennen, wie z.B. einer TV- oder DVD-Fernbedienung oder einem anderen Edison-Roboter. Wenn du den Infrarotempfänger auf diese Weise benutzt, kannst du mit deinem Roboter Nachrichten senden oder empfangen.

Woran liegt das?

TV-Fernbedienungen verwenden Infrarotlicht, um 'Nachrichten' an den Fernseher zu senden. Diese Nachrichten sind jedoch keine Nachrichten, wie man sie vielleicht an einen Freund sendet. Die Nachricht ist ein voreingestelltes Signal, das eine bestimmte Funktion aktiviert. Eine Nachricht weist zum Beispiel den Fernseher an, die Lautstärke zu erhöhen, eine andere Nachricht weist den Fernseher an, die Lautstärke zu verringern.

Jede Taste auf einer Fernbedienung sendet mit Infrarotlicht eine andere Nachricht!

Name_____

Wir können auch Infrarotlicht benutzen, um Nachrichten mit Edison-Robotern zu senden und zu empfangen. Ein Roboter kann mit seinen beiden IR-LEDs, die der IR-Empfänger eines anderen Roboters erkennen kann, eine Infrarot-Nachricht aussenden.

Nicht vergessen

Infrarot wird manchmal als IR abgekürzt. In EdScratch sind IR-Meldungsblöcke Blöcke, die sich auf Edison's Infrarot-Meldungen mit Hilfe der Infrarot-LEDs und des Infrarot-Empfängers beziehen.

Aufgabe 2: Nachricht empfangen

Um den Nachrichtenaustausch mit Edison-Robotern zu nutzen, brauchst du immer mindestens zwei Roboter, einen zum Versenden der IR-Nachrichten und einen zum Erkennen und Reagieren auf die IR-Nachrichten.

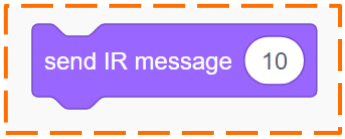
Für diese Aktivität brauchst du einen Partner oder eine Gruppe. Ein Roboter wird eine Nachricht verschicken. Alle anderen Roboter müssen warten, bis sie eine Nachricht erkennen. Sobald die empfangenden Roboter eine Nachricht erkennen, sollte jeder Roboter mit Tanzen reagieren!

Woran liegt das?

Du kannst Edisons Infrarotsensor in einem Programm verwenden, das Edison anweist, ein IR-Meldungsereignis zu erkennen. Du brauchst aber auch einen anderen Edison-Roboter, der ein Programm ausführt, das eine IR-Meldung aussendet, sonst hat dein erster Roboter nichts zu erkennen!

Das Programm „sending IR message“ (IR-Nachricht senden).

Um eine Infrarot-Nachricht mit deinem Edison-Roboter in EdScratch zu senden, musst du diesen Block benutzen:



Name_____

1. Der Block sending IR message ist in der Kategorie LED in EdScratch. Warum ist das so?

_____

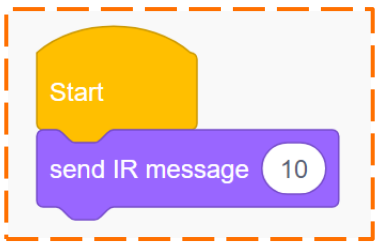
_____

Der sending IR message-block hat einen Eingabeparameter, den du so einstellen kannst, dass eine bestimmte Nachricht gesendet wird.

Woran liegt das?

Genauso wie eine Fernbedienung verschiedene Nachrichten an einen Fernseher senden kann, kannst du von deinem Edison-Roboter verschiedene Nachrichten senden. In EdScratch kannst du die Nachricht ändern, indem du den Wert des Eingabeparameters im sending IR message-block änderst. Der sending IR message-block hat einen Eingabebereich von 0 bis 255. Mit anderen Worten: Edison kann 256 verschiedene 'Nachrichten' senden und empfangen. Stell dir vor, eine Fernbedienung könnte so viele Nachrichten senden. Das wären VIELE Knöpfe!

Manchmal wollen wir Programme erstellen, die Edison anweisen, auf eine bestimmte Art und Weise auf eine bestimmte Nachricht zu reagieren. Andere Male, wie in dieser Aktivität, wollen wir einfach nur, dass Edison reagiert, wenn eine IR-Nachricht erkannt wird. Der Roboter, der die IR-Meldung an die anderen Roboter sendet, um sie zu erkennen, muss dieses Basisprogramm verwenden:

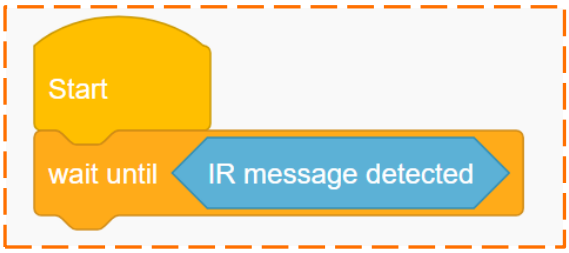


Für diese Aktivität kannst du den Eingabeparameter im Block sending IR message auf einen beliebigen Wert setzen. Du musst auch mehr Code für diesen Roboter schreiben, der nach dem Senden der IR-Meldung ausgeführt werden soll. Benutze mindestens zwei verschiedene Arten von Ausgaben, einschließlich Blöcken aus der Kategorie Drive, um deinen Roboter tanzen zu lassen, nachdem er die IR-Meldung gesendet hat.

Name_____

Das Programm „receiving IR message“ (Empfange IR-Nachricht)

Alle Roboter, die die IR-Meldung erkennen sollen, müssen dasselbe Basisprogramm haben:



Dieses Programm benutzt einen Block aus der Sensing Kategorie in EdScratch mit dem wait until Kontrollblock. Du brauchst keinen Block, um die Erkennung der IR-Nachricht einzuschalten.

Woran liegt das?

Einige von Edisons Sensoren sind immer eingeschaltet und prüfen auf Ereignisse. Der Infrarotempfänger ist einer dieser 'immer an' Sensoren, du brauchst also keinen Code, um ihn einzuschalten. Du musst ihm allerdings sagen, welche Art von IR-Ereignis er erkennen soll und was er tun soll, wenn dieses Ereignis erkannt wird.

Im Beispiel des Basisprogramms weist der Code Edison an, nach einem IR-Ereignis zu suchen. Dieses Ereignis wird ausgelöst, wenn der Roboter eine IR-Meldung entdeckt, unabhängig davon, welche Meldung er entdeckt. Deshalb ist es auch egal, welchen Wert das Programm 'IR-Nachricht senden' verwendet!

Name_____

Benutze das Basisprogramm und schreibe mehr Code, um dem Roboter zu sagen, was er tun soll, sobald er eine IR-Meldung erhält. Benutze mindestens zwei verschiedene Arten von Ausgängen, einschließlich Blöcken aus der Kategorie Drive, um deinen Roboter zum Tanzen zu bringen, sobald er die IR-Meldung erhält.

Sobald alle Roboter ihre Programme heruntergeladen haben, ist es Zeit zum Feiern!

Tipp!

Drücke zuerst auf allen empfangenden Robotern die Play-Taste (Dreieck), dann auf dem sendenden Roboter. Wenn du das Programm des sendenden Roboters ausführst, bevor die empfangenden Roboter bereit sind, werden die empfangenden Roboter die Nachricht verpassen!

Name_____

U4-2.5a: Ferngesteuerte Flaggenmaschine

Erinnerst du dich an Edisons spezielle Strichcodes? Edison kommt mit einigen bereits geladenen Programmen, auf die wir den Roboter mit verschiedenen Strichcodes zugreifen lassen können. Es gibt auch eine Reihe von Strichcodes, die wir zusammen mit Edisons Infrarotsensor benutzen können, um den Roboter mit einer TV- oder DVD-Fernbedienung zu koppeln. Wir können dann ein EdScratch-Programm schreiben, das Edison sagt, was er tun soll, wenn er einen bestimmten Fernbedienungscode erkennt.

Woran liegt das?

Die programmierbaren TV-Fernbedienungscodes sind eine besondere Art von Strichcode. Mit diesen Strichcodes kann Edison einen bestimmten Code speichern, den der Roboter später referenzieren kann.

Im Gegensatz zu anderen Edison-Strichcodes aktivieren diese Strichcodes kein Programm in Edison. Stattdessen weisen die Strichcodes Edison an, den nächsten Tastendruck auf der Fernbedienung zu speichern, den der Roboter als einen bestimmten Fernbedienungscode erkennt.

Um einen programmierbaren TV-Fernbedienungscode zu verwenden, musst du zuerst den Strichcode mit deinem Edison scannen und ihn mit einer Taste deiner Wahl auf einer Fernbedienung verbinden. Du könntest zum Beispiel den Fernbedienungscode #1 einscannen und ihn mit der Taste 'Lautstärke erhöhen' auf deiner Fernbedienung verbinden. Sobald die Verbindung hergestellt ist, wird Edison das IR-Signal als 'Fernbedienungscode #1' referenzieren, sobald du diese Taste auf der Fernbedienung drückst.

Du kannst dann ein Programm in EdScratch schreiben, das den gleichen Code als Eingabe verwendet. Dein EdScratch-Programm muss Edison mitteilen, welche Fernbedienungscodes erkannt werden sollen und was zu tun ist, wenn der Roboter einen bestimmten Code erkennt.

Indem wir die Fernbedienungscodes als Ereignisse in EdScratch-Programmen verwenden, kannst du Roboterkreationen bauen, die du mit einer TV-Fernbedienung steuern kannst! Lass uns versuchen, eine Flaggenmaschine mit Edisons Motorausgängen zu bauen, die du mit einer TV-Fernbedienung steuern kannst.

Name_____

Was zu tun ist

Das Ziel dieser Aktivität ist es, eine Flaggenmaschine zu bauen und zu programmieren, mit der du dazu beitragen kannst, dass das Lernen für einen Test mehr Spaß macht.

Deine Kreation sollte es dir ermöglichen, eine zweiseitige Flagge zu kontrollieren, die auf der einen Seite das Wort 'Ja' und auf der anderen Seite das Wort 'Nein' hat. Du musst ein Programm in EdScratch schreiben, mit dem du die 'Ja'-Seite der Flagge anzeigen kannst, wenn du eine Taste auf einer Fernbedienung drückst und die 'Nein'-Seite, wenn du eine andere Taste drückst. Wenn eure Flaggenmaschine fertig ist, arbeitet mit einem Partner zusammen und fragt euch abwechselnd gegenseitig ab. Benutzt eure Flaggenmaschine, um zu signalisieren, ob euer Partner eure Frage richtig beantwortet hat oder nicht!

Du musst entwerfen, wie deine Flaggenmaschine funktioniert, indem du Edisons Ausgänge benutzt und ein Programm in EdScratch schreibst, um sie mit Ferncodes zu steuern.

Als erstes müsst ihr euren Roboter mit einer Fernsteuerung koppeln. Schau dir die programmierbaren TV-Fernbedienungscodes auf dem Arbeitsblatt U4-6 an. Für diese Aktivität wirst du wahrscheinlich zwei dieser Fernbedienungscodes verwenden müssen.

Um euren Roboter mit einem dieser Codes zu koppeln, müsst ihr die folgenden Schritte befolgen:

1. Platziere Edison gegenüber dem Strichcode auf der rechten Seite des Strichcodes.
2. Drücke dreimal die Aufnahmetaste (rund).
3. Warte, während Edison vorwärts fährt und den Strichcode einscannt.
4. Wähle eine Taste auf deiner Fernbedienung aus, die mit diesem Fernbedienungscode übereinstimmen soll. Richte die Fernbedienung auf deinen Roboter und drücke die ausgewählte Taste. Achte darauf und benutze für jeden Fernbedienungscode eine andere Taste der Fernbedienung.

Tipp!

Du brauchst Edison, um fahren zu können, um einen Strichcode einscannen zu können, also stell sicher, dass du alle Strichcodes einscannst, die du brauchst, bevor du mit dem Bau deiner Roboterkreation beginnst!

Name_____

Du musst auch ein Programm in EdScratch schreiben, das Edison sagt, was es tun soll, wenn es die verschiedenen Fernbedienungscodes erkennt. Dein Programm sollte Edison anweisen, eine Seite der Flagge zu zeigen, wenn du die erste Fernbedienungstaste deines Pairings drückst und die andere Seite der Flagge, wenn du die andere Taste drückst.

Wenn du deine Kreation gebaut und programmiert hast, probiere sie aus!

Name_____

U4-2.5b: Baue und steuere den EdCrane

Edison hat eine Reihe von Strichcodes, die wir zusammen mit Edisons Infrarotsensor benutzen können, um den Roboter mit einer TV- oder DVD-Fernbedienung zu koppeln. Wir können dann ein EdScratch-Programm schreiben, das Edison sagt, was er tun soll, wenn er einen bestimmten Fernbedienungscode erkennt.

Woran liegt das?

Die programmierbaren TV-Fernbedienungscodes sind eine besondere Art von Strichcode. Mit diesen Strichcodes kann Edison einen bestimmten Code speichern, den der Roboter später referenzieren kann.

Im Gegensatz zu anderen Edison-Strichcodes aktivieren diese Strichcodes kein Programm in Edison. Stattdessen weisen die Strichcodes Edison an, den nächsten Tastendruck auf der Fernbedienung zu speichern, den der Roboter als einen bestimmten Fernbedienungscode erkennt.

Um einen programmierbaren TV-Fernbedienungscode zu verwenden, musst du zuerst den Strichcode mit deinem Edison scannen und ihn mit einer Taste deiner Wahl auf einer Fernbedienung verbinden. Du könntest zum Beispiel den Fernbedienungscode #1 einscannen und ihn mit der Taste 'Lautstärke erhöhen' auf deiner Fernbedienung verbinden. Sobald die Verbindung hergestellt ist, wird Edison das IR-Signal als 'Fernbedienungscode #1' referenzieren, sobald du diese Taste auf der Fernbedienung drückst.

Du kannst dann ein Programm in EdScratch schreiben, das den gleichen Code als Eingabe verwendet. Dein EdScratch-Programm muss Edison mitteilen, welche Fernbedienungscodes erkannt werden sollen und was zu tun ist, wenn der Roboter einen bestimmten Code erkennt.

Indem wir die Fernbedienungscodes als Ereignisse in EdScratch-Programmen verwenden, kannst du Roboterkreationen bauen, die du mit einer TV-Fernbedienung steuern kannst!

Was zu tun ist

In dieser Aktivität wirst du den EdCrane bauen und programmieren. Der EdCrane ist ein ferngesteuerter Kran mit einem magnetischen 'Haken', mit dem du kleine Metallgegenstände heben und bewegen kannst.

Als erstes musst du deinen Roboter mit Hilfe der programmierbaren TV-Fernbedienungscodes auf Arbeitsblatt U4-6 mit einer Fernbedienung koppeln. Für diese

Name_____

Aktivität müsst ihr vier Fernbedienungscodes verwenden, um dem EdCrane zu sagen, dass er eine von vier verschiedenen Aktionen ausführen soll.

1. Plane deine Fernbedienungscodes und Fernbedienungstasten ein:

EdCrane-Action	Code der Fernbedienung	Fernbedienungstaste
Drehe die Magnetspule im Uhrzeigersinn		
Drehe die Magnetspule gegen den Uhrzeigersinn		
Drehe den Kran im Uhrzeigersinn		
Drehe den Kran gegen den Uhrzeigersinn		

Benutzt euren Plan als Referenz und koppelt euren Roboter an jeden Code. Um euren Roboter mit einem dieser Codes zu koppeln, müsst ihr die folgenden Schritte befolgen:

1. Platziere Edison gegenüber dem Strichcode auf der rechten Seite des Strichcodes.
2. Drücke dreimal auf die Aufnahmetaste (rund).
3. Warte, während Edison vorwärts fährt und den Strichcode einscannt.
4. Wähle eine Taste auf deiner Fernbedienung aus, die mit diesem Fernbedienungscode übereinstimmen soll. Richte die Fernbedienung auf deinen Roboter und drücke die ausgewählte Taste. Achte darauf und benutze für jeden Fernbedienungscode eine andere Taste der Fernbedienung.

Tipp!

Du brauchst Edison, um fahren zu können, um einen Strichcode einscannen zu können, also stell sicher, dass du alle Strichcodes einscannst, die du brauchst, bevor du mit dem Bau deiner Roboterkreation beginnst!

Sobald der Roboter mit der Fernbedienung gepaart ist, kannst du den EdCrane bauen und ein Programm in EdScratch schreiben, das Edison sagt, was es tun soll, wenn es jeden der vier verschiedenen Fernbedienungscodes erkennt.

Name_____

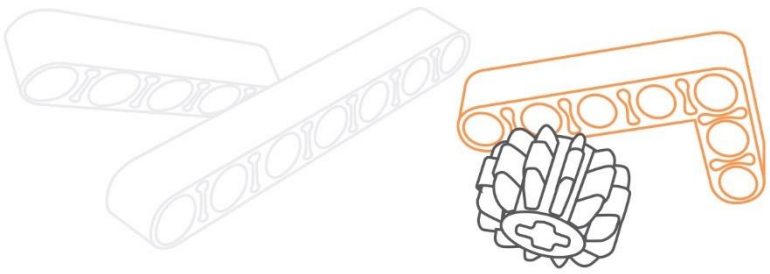
Benutze diesen Link

Gehe zu meetedison.com/Inhalt/EdCreate/EdBuild-EdCrane-instructions.pdf

Dieser Link führt euch zu den Schritt-für-Schritt-Anleitungen für den Bau des EdCrane.

Benutze deinen Plan als Referenz und schreibe dein EdScratch-Programm so, dass der EdCrane auf jeden Remote-Code mit den Ausgaben reagiert, die er benötigt, um die gewünschte Aktion auszuführen.

Wenn du deine Kreation gebaut und programmiert hast, probiere sie aus!



Name_____

U4-2.5c: Wasserwerfer zur Brandbekämpfung

Brände zu bekämpfen, einschließlich großer Katastrophen wie Waldbrände, ist eine gefährliche, aber notwendige Aufgabe. Wenn schreckliche Brände ausbrechen, können die Bedingungen für die Menschen extrem gefährlich werden. Eine Möglichkeit, bei der Bekämpfung dieser Brände zu helfen, ist der Einsatz von Feuerlöschrobotern.

Für diese Herausforderung musst du den EdTank so bauen und programmieren, dass er mit seiner Kanone einen Wasserstrahl (repräsentiert durch das Gummiband) abschießt, um bei der Brandbekämpfung zu helfen.

Was zu tun ist

Um dieses Projekt abzuschließen, müsst ihr den EdTank bauen, euren Testraum erstellen und euren EdTank mit EdScratch programmieren.

Einrichtung des Testraums

Ihr müsst den Testraum erstellen, in dem ihr euer Löschwasserwerfer-Programm laufen lassen wollt. Erstelle zwei parallele schwarze Linien auf einer weißen Fläche, z.B. einem großen Poster, einem Tisch oder dem Boden. Du kannst die Linien so weit voneinander entfernt platzieren, wie du willst.

Eine schwarze Linie repräsentiert den Stützpunkt des Löschtrupps und die andere die 'Aschelinie', hinter der das Feuer brennt.

Tipp!

Die feuerlöschenden EdTank-Roboter feuern ihre Gummibänder aus der 'Aschelinie' heraus. Achte darauf, dass diese Linie irgendwo hinzeigt, wo niemand getroffen wird, wie zum Beispiel eine Wand.

Name_____

Das EdScratch-Programm

Ihr müsst zwei verschiedene Programme schreiben, eines für den oberen Edison und eines für den unteren Edison im EdTank.

Eure Programme müssen den EdTank so steuern, dass der Roboter vom Stützpunkt des Löschtrupps fährt, bis er auf die 'Aschelinie' trifft (aber nicht weiter), seine Wassernutzlast abschießt (also die Kanone abfeuert) und dann zum Stützpunkt des Löschtrupps zurückkehrt.

Tipp!

- Sie müssen einen programmierbaren TV-Fernbedienungscode aus dem Aktivitätsblatt U4-6 verwenden.
- Sie können Messaging in Ihren Programmen verwenden, damit die beiden Edison-Roboter miteinander kommunizieren
- Pseudocode hilft uns bei der Planung, Kommentare helfen uns beim Debuggen. Verwenden Sie sie, um Ihr Programm zu planen und zu testen!

Der Bau

Ihr müsst den EdTank bauen, um ihn als Wasserwerfer zur Brandbekämpfung zu verwenden.

Benutze diesen Link

Gehe zu meetedison.com/Inhalt/EdCreate/EdBuild-EdTank-instructions.pdf

Dieser Link führt euch zu den Schritt-für-Schritt-Anleitungen für den Bau des EdTanks.

Denkt daran, dass jeder der beiden Roboter mit einem eurer EdScratch-Programme programmiert werden muss. Stelle sicher, dass du das richtige Programm in den richtigen Roboter bekommst!

Name_____

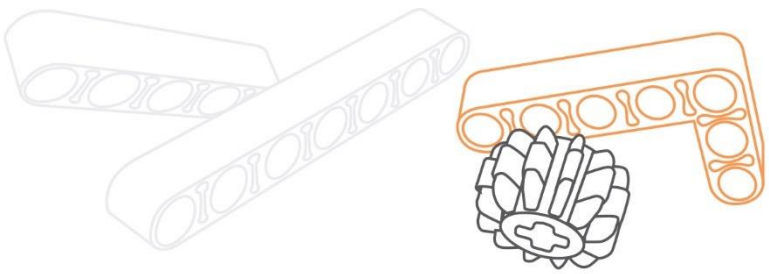
Probiere es aus!

Probiere deine Lösung auf dem Testplatz aus.

Ist dein Feuerwehrroboter bis zur 'Aschelinie' gefahren, hat die Wasserkanone abgefeuert und ist dann zurück zur Basis gekommen, um wieder geladen zu werden?

Tipp!

Wenn dein Feuerwehrroboter nicht ganz richtig funktioniert, gib nicht auf! Experimentiere, teste und löse weiter Probleme, bis dein Feuerwehrroboter wie ein Held funktioniert!



Name_____

U4-2.5d: Halbautomatischer Bagger

Ferngesteuerte Baumaschinen ermöglichen es den Menschen, potenziell gefährliche Aufgaben oder Umgebungen sicherer zu bewältigen. Bei Bauprojekten, die einen Abriss erfordern, ist es besonders wichtig, die Maschinenführer von herumfliegenden Trümmern fernzuhalten. Viele Aufgaben auf einer Baustelle müssen immer wieder ausgeführt werden. Durch die Voreinstellung oder Automatisierung eines Teils dieser Art von Aufgaben wird die Anzahl der aktiven Operationen, die der Kontrolleur ausführen muss, begrenzt. Durch die Automatisierung wird Zeit gespart und die Wahrscheinlichkeit menschlicher Fehler begrenzt, während gleichzeitig die Sicherheit der Bediener erhöht wird.

Für diese Herausforderung musst du den EdDigger so bauen und programmieren, dass er eine Trümmeraufräumung halbautomatisch durchführt.

Was zu tun ist

Um dieses Projekt zu vollenden, musst du den EdDigger bauen, deinen Testplatz erstellen und deinen EdDigger mit EdScratch programmieren.

Einrichtung des Testplatzes

Du musst den Testraum erstellen, den du für dein Trümmerbeseitigungsprogramm benutzen wirst. Erstelle eine 'Baustelle' mit einem Haufen 'Schutt' in einem Bereich und einer Drop-off-Zone in einem anderen Bereich, wo der Schutt angeliefert werden muss, damit er weggeschafft werden kann.

Tipp!

Du kannst beliebige kleine Gegenstände benutzen, um die Trümmer darzustellen, sogar zusätzliche Stücke von EdCreate! Stelle nur sicher, dass der EdDigger die Objekte, die du auswählst, vergrößern kann.

Das EdScratch-Programm

Du wirst zwei verschiedene Programme schreiben müssen, eines für den oberen Edison und eines für den unteren Edison im EdDigger. Der Roboter sollte sich nur dann in Bewegung setzen, wenn der Bediener der Fernsteuerung den richtigen Knopf auf einer Fernbedienung drückt.

Sobald der Fernbedienungscode erkannt wird, sollte der Roboter in der Lage sein, die Aufgabe der Trümmerbeseitigung selbstständig durchzuführen. Eure Programme müssen den EdDigger so steuern, dass der Roboter von der Abwurfzone zum Schutthaufen fährt, etwas Schutt aufschaufelt, dann zur Abwurfzone zurückkehrt und den Schutt abliefert.

Name_____

Tipp!

- Ihr müsst einen programmierbaren TV-Fernbedienungscode aus dem Aktivitätsblatt U4-6 verwenden.
- Du kannst Messaging in deinen Programmen verwenden, damit die beiden Edison-Roboter miteinander kommunizieren können.
- Pseudocode hilft uns bei der Planung, Kommentare helfen uns beim Debuggen. Benutze sie, um dein Programm zu planen und zu testen!

Die Konstruktion

Ihr müsst den EdDigger bauen, um ihn bei eurer Aufgabe der Trümmerbeseitigung zu verwenden.

Benutze diesen Link

Gehe zu meetedison.com/Inhalt/Erstelle/EdBuild-EdDigger-Anweisungen.pdf

Dieser Link führt euch zu den Schritt-für-Schritt-Anleitungen für den Bau des EdDiggers.

Denke daran, dass jeder der beiden Roboter mit einem deiner EdScratch-Programme programmiert werden muss. Stelle sicher, dass du das richtige Programm in den richtigen Roboter bekommst!

Name_____

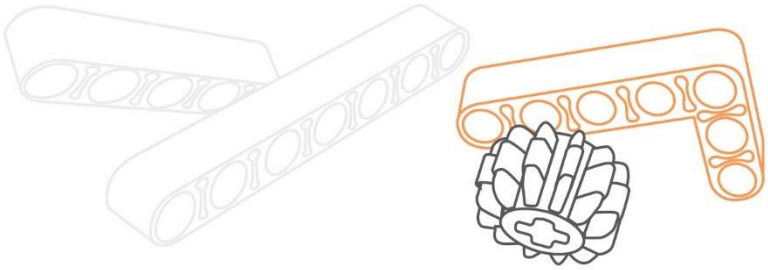
Probiere es aus!

Teste deine Lösung auf dem Testplatz aus.

Hat dein Roboter auf einen ferngesteuerten Code gewartet, fährt dann zum Schutthaufen, schaufelt etwas Schutt auf, bringt ihn dann zurück zur Ablagezone und liefert ihn ab?

Tipp!

Wenn dein Trümmeraufräumer nicht ganz richtig arbeitet, gib nicht auf! Experimentiere weiter, teste und löse Probleme, bis dein Roboter wie der Nr. 1 Mitarbeiter arbeitet!



Name_____

U4-2.5e: Entfernung von gefährlichem Material

Viele Arten von Materialien können für Menschen gefährlich sein, wie zum Beispiel biomedizinische Abfälle, Nebenprodukte der Herstellung oder radioaktives Material. Um die Sicherheit der Menschen zu gewährleisten, müssen diese Arten von Materialien sicher von bewohnten Gebieten weggebracht und entsorgt werden. Ferngesteuerte Fahrzeuge mit Roboterarmen sind eine Möglichkeit, diese Aufgabe zu bewältigen.

Für diese Herausforderung müsst ihr den EdRoboClaw so bauen und programmieren, dass er mit seinem Gelenkarm gefährliche Stoffe abtransportiert und in einem vorher festgelegten Bereich ablegt.

Was ihr tun müsst

Um dieses Projekt abzuschließen, müsst ihr EdRoboClaw bauen, euren Testraum erstellen und euer EdRoboClaw mit EdScratch programmieren.

Einrichtung des Testraums

Du musst den Testraum erstellen, den du für dein Sondermüllbeseitigungsprogramm benutzen wirst. Erstelle eine 'Entsorgungszone', indem du einen Bereich, z.B. ein Quadrat, mit schwarzen Linien auf einer weißen Hintergrundfläche markierst. Die Innenseite der 'Entsorgungszone' ist der Bereich, in dem das gefährliche Material abgeladen werden muss. Du brauchst auch etwas, das den gefährlichen Abfall darstellt.

Tipp!

Stellt sicher, dass der Roboterarm alles aufheben und tragen kann, was den gefährlichen Abfall darstellt.

Das EdScratch-Programm

Du wirst zwei verschiedene Programme schreiben müssen, eines für den oberen Edison und eines für den unteren Edison im EdRoboClaw.

Eure Programme müssen den EdRoboClaw so steuern, dass der Roboter den gefährlichen Abfall aufnimmt und zur Entsorgungszone bringt, wo er den Abfall ablegt. Dein Roboter sollte diese Zone nicht betreten, aber sein Arm kann in die Entsorgungszone reichen. Sobald das Material abgeworfen wurde, sollte sich der Roboter aus der Zone zurückziehen.

Name_____

Tipp!

- Vielleicht möchtt ihr den Roboter mit der Klaue starten, die bereits offen über dem "Abfall" schwebt.
- Ihr könnt Messaging in euren Programmen verwenden, um die beiden Edison-Roboter miteinander kommunizieren zu lassen.
- Pseudocode hilft uns bei der Planung, Kommentare helfen uns bei der Fehlersuche. Verwendet sie, um euer Programm zu planen und zu testen!

Der Bau

Ihr müsst die EdRoboClaw bauen, um sie bei eurer Aufgabe zur Beseitigung von Sondermüll zu verwenden.

Benutze diesen Link

Geht zu meetedison.com/Inhalt/EdCreate/EdBuild-EdRoboClaw-Anweisungen.pdf

Dieser Link führt zu den Schritt-für-Schritt-Anleitungen für den Aufbau des EdRoboClaw.

Denkt daran, dass jeder der beiden Roboter mit einem eurer EdScratch-Programme programmiert werden muss. Stelle sicher, dass du das richtige Programm in den richtigen Roboter bekommst!

Name_____

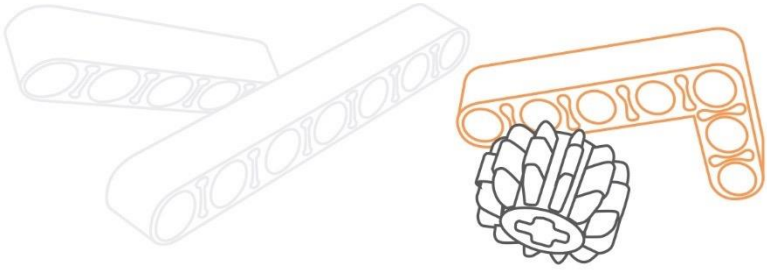
Probiere es aus!

Teste deine Lösung auf dem Testplatz aus.

Hat dein Roboter den Giftmüll eingesammelt, ihn zur Entsorgungszone getragen, den Abfall innerhalb des dafür vorgesehenen Bereichs abgelegt und sich dann von der Entsorgungszone zurückgezogen?

Tipp!

Wenn dein Roboter nicht ganz richtig funktioniert, gib nicht auf! Experimentiere, teste und löse weiter Probleme, bis dein Roboter den Sondermüll entfernt hat und sich wieder in Sicherheit bringt!



Name_____

U4-2.5f: Brieftauben

Brieftauben sind eine Art von Haustauben, die von der Felsentaube abstammen. In freier Wildbahn haben die Felsentauben eine unglaubliche angeborene Fähigkeit: sie können den Weg nach Hause auch aus sehr großen Entfernungen finden. Die Menschen haben Brieftauben für dieses besondere Talent gezüchtet und benutzen die Vögel, um zu ihren Häusern zurückzufliegen und Botschaften mit sich zu tragen!

Kannst du deinen Edison-Roboter dazu bringen, sich wie eine Brieftaube zu verhalten?

Was du tun kannst

Um dieses Projekt abzuschließen, müsst ihr einen Weg finden, Edison dazu zu bringen, sich so sehr wie eine Brieftaube zu verhalten und von woanders zu einer bestimmten 'Heimatbasis' zurückzukehren. Ihr müsst einen Testraum und ein Programm erstellen, das in eurem Edison-Roboter läuft.

Wie du deinen Testraum entwirfst und wie dein Programm funktioniert, bleibt dir überlassen. Du kannst jeden der Edison-Sensoren auf jede beliebige Art und Weise benutzen, einschließlich Linienverfolgung, Hinderniserkennung und IR-Meldungen.

Name_____

Tipp!

- Sensoren wie der Hinderniserkennungsstrahl und der Line Tracker müssen eingeschaltet sein, damit sie funktionieren. Das Programm muss Edison auch sagen, auf welches Ereignis die Sensoren achten sollen und wie sie auf dieses Ereignis reagieren sollen.

- Einige der Sensoren von Edison, einschließlich der Hinderniserkennung, erzeugen und speichern Daten, wenn sie bestimmte Ereignisse erkennen. Es ist gute Codierpraxis, die Sensordaten zu löschen, besonders wenn man Sensorereignisse in Bedingungen verwendet, die in Schleifen verschachtelt sind. Du willst nicht, dass die Daten aus einer vorherigen Schleife die nächste Schleife beeinflussen! Es ist auch ratsam, die Sensordaten beim Start eines Programms zu löschen, nur für den Fall, dass der Roboter alte Daten aus einem früheren Programm gespeichert hatte.

- Einige der Sensoren von Edison, wie IR-Meldungen und Hinderniserkennung, können nicht gleichzeitig verwendet werden. Du kannst die Sensoren jedoch an verschiedenen Stellen in deinem Programm ein- und wieder ausschalten, wenn du willst.

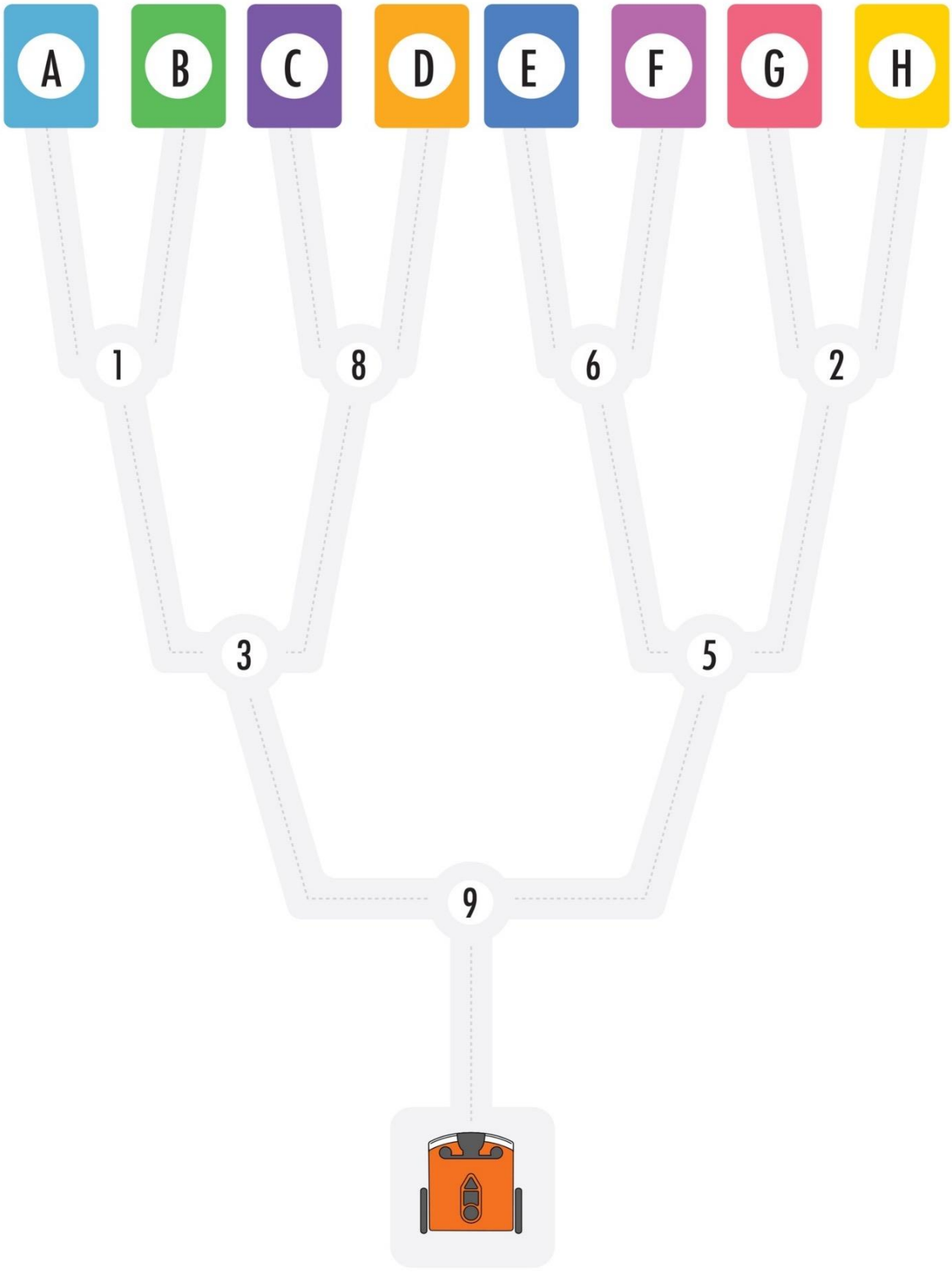
- Nutze deine Testraumgestaltung zu deinem Vorteil. Was könntest du bauen, das Edison helfen würde, zu suchen und wieder nach Hause zu kommen?

- Wie könntest du den Robotern signalisieren, dass es an der Zeit ist, ihren Weg zurück nach Hause zu suchen?

- Pseudocode hilft uns bei der Planung, Kommentare helfen uns beim Debuggen. Benutze sie, um dir zu helfen, dein Programm zu planen und zu testen!

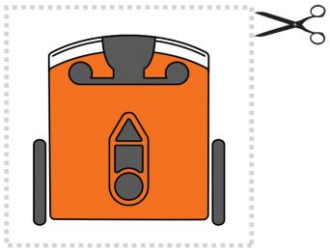
Name_____

Arbeitsblatt U4-1: Wenn-Dann-Labyrinth



Arbeitsblatt U4-2: Pseudocode Schritt-für-Schritt

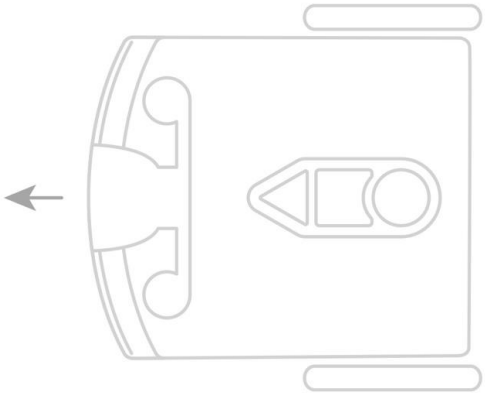
				J		D
	6		I			
F		C	7		9	
		H		4	G	2
3		1				E
B			A	8	5	

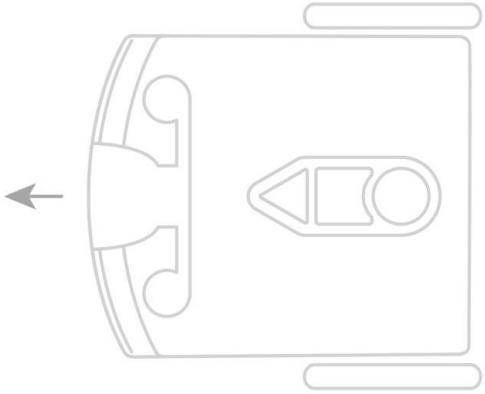


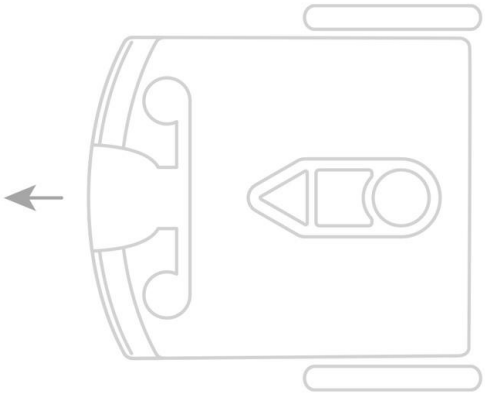
Name _____

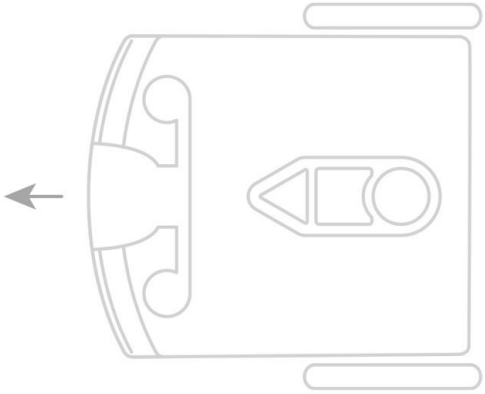
Name_____

Arbeitsblatt U4-3: Line Tracker Testzone

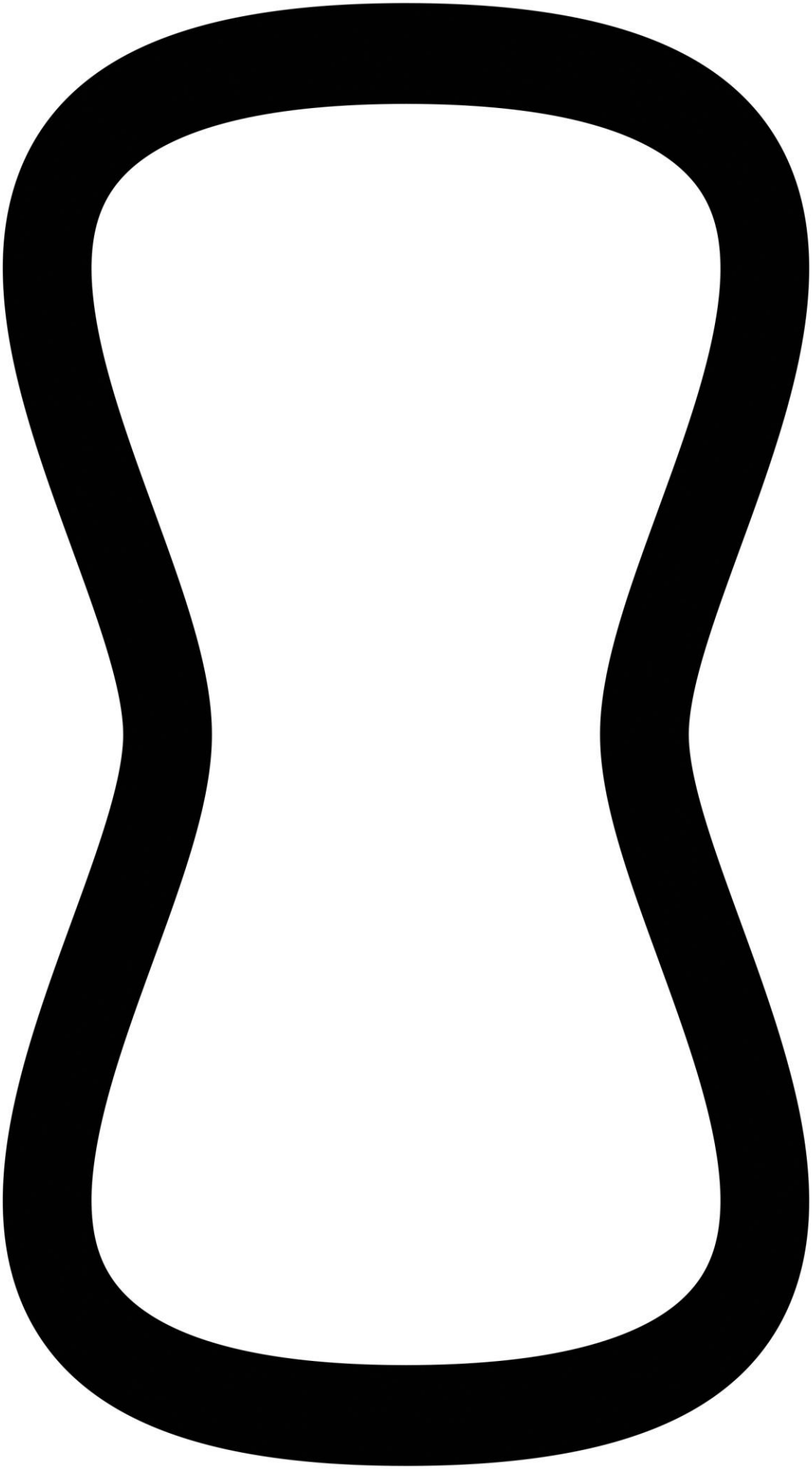








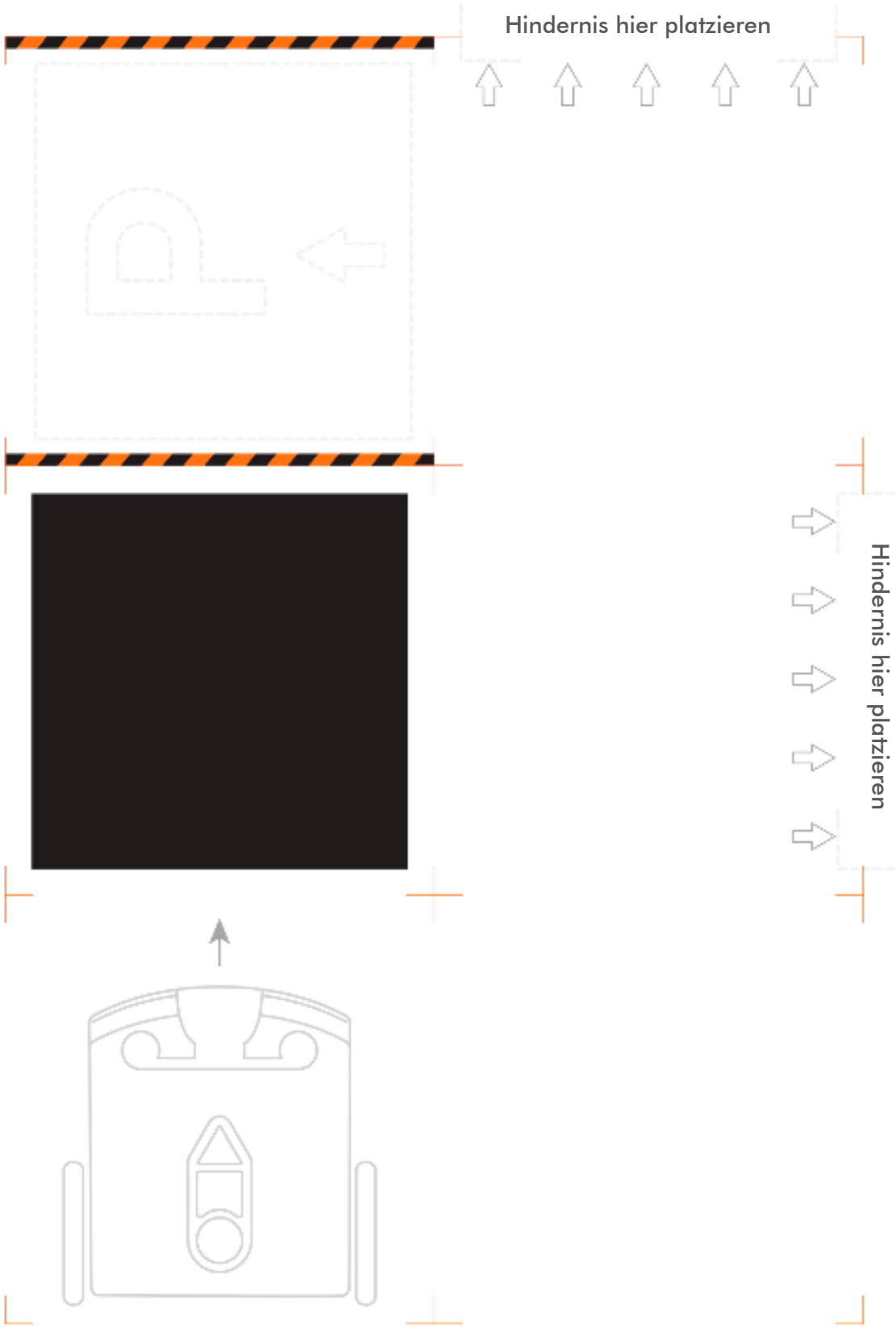
Name_____



Arbeitsblatt U4-4: Nicht reflektierender Rand

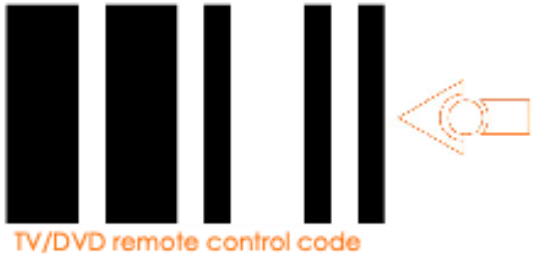
Name_____

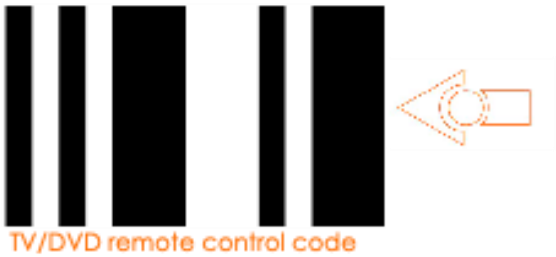
Arbeitsblatt U4-5: Wenn Linie, nach rechts gehen

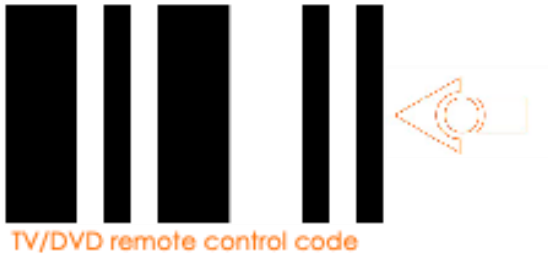


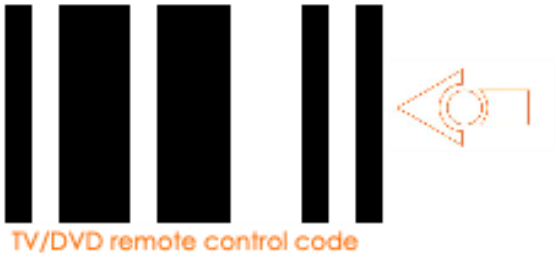
Name_____

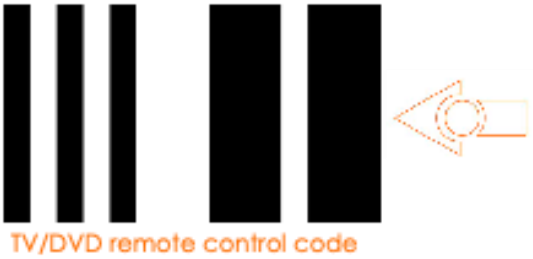
Arbeitsblatt U4-6: Programmierbare Fernbedienungscodes

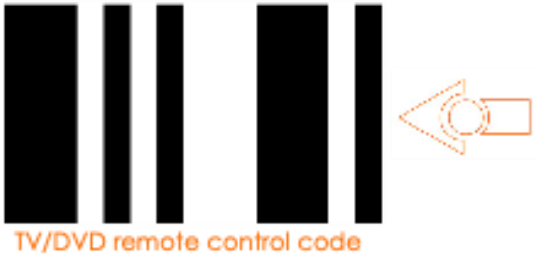




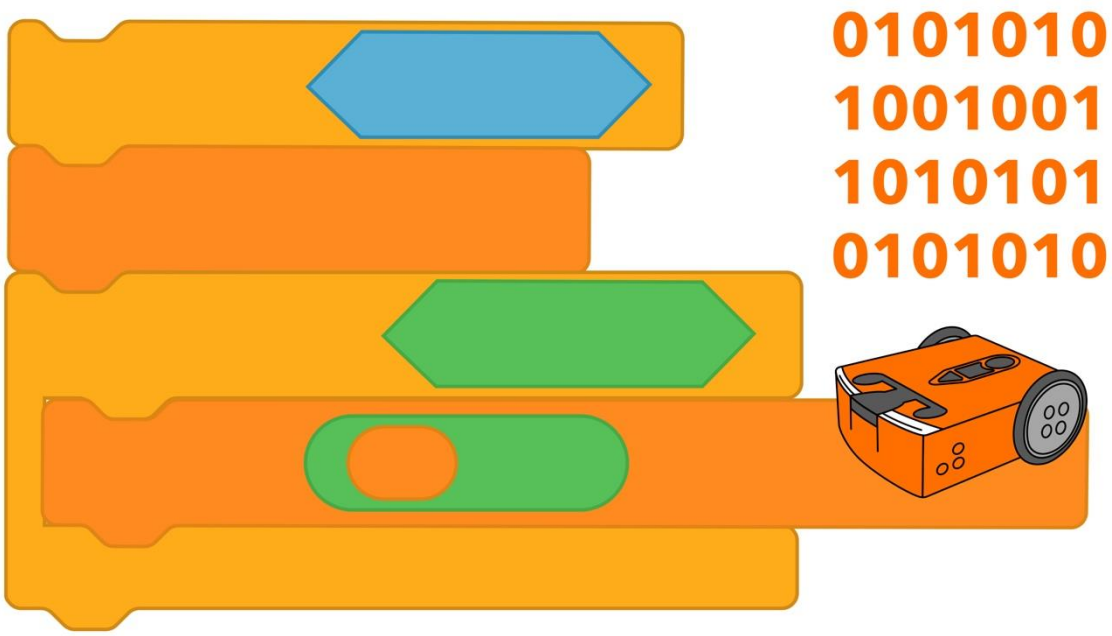








Einheit 5: Vielseitige Variablen



Edison

Name_____

U5-1.1: „Expressions“

Ob du es glaubst oder nicht, Computersprachen sind meistens für Menschen, nicht für Computer.

Woran liegt das?

Computer können eigentlich nur in Zahlen 'denken', nicht in Worten oder Codeblöcken. Zum Beispiel kann Edison die Blöcke in EdScratch nicht so verstehen, wie sie auf deinem Computerbildschirm betrachtet werden. Die Blöcke müssen in ein Format gebracht werden, das Edison verstehen kann, bevor das Programm heruntergeladen werden kann. Dieser Prozess, der ein wenig Zeit in Anspruch nehmen kann, ist der Grund, warum es ein wenig dauern kann, bis der Programm-Edison-Button im Pop-up-Fenster erscheint, wenn du ein Programm von EdScratch auf Edison herunterlädst.

Computersprachen, wie EdScratch, machen es uns viel einfacher, Programme zu erstellen. Ohne eine Computersprache müsstest du jeden einzelnen Befehl mit nichts anderem als 1er und 0er schreiben!

Die Blöcke in EdScratch machen es viel einfacher für uns, Edison die Eingaben zu geben, die wir wollen und Programme zu schreiben, die Edison sagen, was zu tun ist.

Es ist jedoch wichtig, daran zu denken, dass Edison den ganzen Code, den wir programmieren, als Zahlen sieht. Gleichermaßen, wenn Edison Daten von einem Sensor in seinem Speicher ablegt, werden diese Daten in Form von Zahlen gespeichert.

Nicht vergessen

Eingaben sind die Informationen und Anweisungen, die du einem Computer gibst.

Wir können Zahlen in den Programmen, die wir schreiben, auf verschiedene Weise verwenden. Wenn wir in Computerprogrammen Zahlen und ein bisschen Mathematik verwenden können, können wir den Roboter dazu bringen, viele verschiedene Dinge zu tun. Eine Art und Weise, wie wir Zahlen und Mathematik in EdScratch verwenden können, ist in „expressions“.

Name_____

Fachjargon

Im Code ist ein Ausdruck eine Frage, die entweder als 'wahr' oder 'falsch' ausgewertet und gelöst werden kann. Ausdrücke können mit anderem Code verwendet werden, insbesondere mit bedingtem Code wie Till-Blöcken oder If-Blöcken, um den Ablauf eines Programms zu steuern.

Mit Ausdrücken können wir zwei Zahlen oder Datenbits miteinander vergleichen. Je nach Ergebnis können wir Edison dann sagen, was er tun soll.

Um Ausdrücke in einem EdScratch Programm verwenden zu können, musst du wissen, wie man Ausdrücke liest, auswertet und auflöst, so wie es ein Computer tut.

Ausdrücke auflösen

In EdScratch enthält die Kategorie Operators die speziellen Blöcke, die du brauchst, um einen Ausdruck zu schreiben. Diese Blöcke verwenden mathematische Notationen (in anderen Worten: Symbole), um die linke Seite mit der rechten Seite des Ausdrucks zu vergleichen.

Zum Beispiel: 'Ist A kleiner als B?' wird in EdScratch als 'A < B' geschrieben und 'Ist A gleich B?' wird als 'A = B' geschrieben.

Schau dir die Liste der Ausdrücke an, die du in EdScratch schreiben kannst:

Expression	Bedeutung
A = B	Ist A dasselbe wie B?
A ≠ B	Ist A nicht gleich B?
A > B	Ist A größer als B?
A >= B	Ist A größer oder gleich B?
A < B	Ist A kleiner als B?
A <= B	Ist A kleiner oder gleich B?

Du kannst das 'A' und 'B' in den Ausdrücken durch einen beliebigen Wert ersetzen. Du kannst auch Berechnungen mit diesen Werten durchführen. Zum Beispiel, '(A + 2) > B' bedeutet 'Ist A plus 2 größer als B?

Im Code arbeiten die Ausdrücke in einer bestimmten Reihenfolge. Wenn dein Ausdruck Berechnungen enthält, wird der Ausdruck die Berechnungen zuerst ausführen. Dann

vergleicht er die linke Seite des Ausdrucks mit der rechten Seite und entscheidet sich entweder für 'wahr' oder 'falsch'.

Weil Ausdrücke Zahlen und manchmal auch Berechnungen enthalten, ist es verlockend, sie wie mathematische Probleme zu betrachten, die eine Antwort haben, die eine Zahl ist. Aber du musst über Ausdrücke nachdenken wie ein Computer. Als erstes solltest du den Ausdruck bewerten, indem du alle Berechnungen auf beiden Seiten des Ausdrucks durchführst. Dann löst du den Ausdruck auf, indem du die linke Seite der Notation mit der rechten Seite vergleichst. Mit anderen Worten, beantworte die Frage, die der Ausdruck stellt, entweder als 'wahr' oder 'falsch'.

Woran liegt das?

Auch wenn Ausdrücke Zahlen bewerten und Mathematik enthalten können, lösen sie immer nur eine von zwei Antworten auf: entweder 'wahr' oder 'falsch'. Deshalb sind sie so hilfreich, wenn sie mit bedingtem Code verwendet werden. Anstatt uns Gedanken darüber machen zu müssen, wie der genaue Wert sein wird, können wir innerhalb von Mengen von 'wahr' und 'falsch' arbeiten.

Nehmen wir zum Beispiel an, du möchtest, dass ein Programm etwas so lange tut, wie ein Wert größer als 10 ist. Anstatt einen Haufen verschachtelter if-Blöcke zu schreiben, um den genauen Wert zu überprüfen, kannst du einfach einen Ausdruck verwenden, der X > 10 sagt. Das Programm wird diesen Wert überprüfen, egal auf welchen Wert X gesetzt ist, und solange X größer als 10 ist, ist die Bedingung erfüllt!

Es gibt einen speziellen Namen in der Informatik für Daten, die nur einen von zwei möglichen Werten wie diesen haben: wir nennen diese Art von Daten Boolesche Daten.

Wenn du in der Lage bist, zu verstehen, was Ausdrücke bedeuten und was sie auflösen, wird es dir helfen, dem Geschehen in einem Programm zu folgen, indem du es einfach 'liest'. Dies ist eine wichtige Fertigkeit im Programmieren, bekannt als „tracing“ (Nachverfolgung).

Name_____

Fachjargon

Code zurückverfolgen bedeutet, ein Programm Zeile für Zeile abzuarbeiten und wichtige Werte aufzuzeichnen. Durch ein Programm zu verfolgen und zu verstehen, was passiert, lässt einen Programmierer herausfinden, wie sein Code laufen soll, auch wenn es viele verschiedene Werte innerhalb des Programms gibt. Das Zurückverfolgen von Code kann helfen, logische Fehler oder Bugs im Code zu suchen, aber es ist auch nützlich, wenn du nur verstehen musst, was in einem Programm passiert.

Wenn du Programme verwendest, die Werte haben und Berechnungen mit diesen Werten anstellst, nimm dir einen Moment Zeit, das Programm zu durchlaufen, um sicherzustellen, dass du verstehst, was vor sich geht. Dies kann dir helfen zu verstehen, was im Code passieren sollte und dein Programm zu debuggen, wenn die Dinge nicht so funktionieren, wie sie sollten.

Probiere es aus!

Versuche die folgenden Ausdrücke aufzulösen. Schreibe zuerst auf, was jeder Ausdruck bedeutet, dann löse ihn entweder in wahr oder falsch auf.

Bei diesen Fragen, wenn A = 2 und B = 4, was bedeutet jeder der folgenden Ausdrücke (mit anderen Worten, welche Frage wird gestellt) und was bedeutet jeder davon (wahr oder falsch)?

1. (A*2) = B

   Bedeutung: _____

   Löst sich auf in: _____

2. A >= B

   Bedeutung: _____

   Löst sich auf in: _____

3. (A+A) ≠ B

   Bedeutung: _____

   Löst sich auf in: _____

4. (A-1) < (B-3)

   Bedeutung: _____

   Löst sich auf in: _____

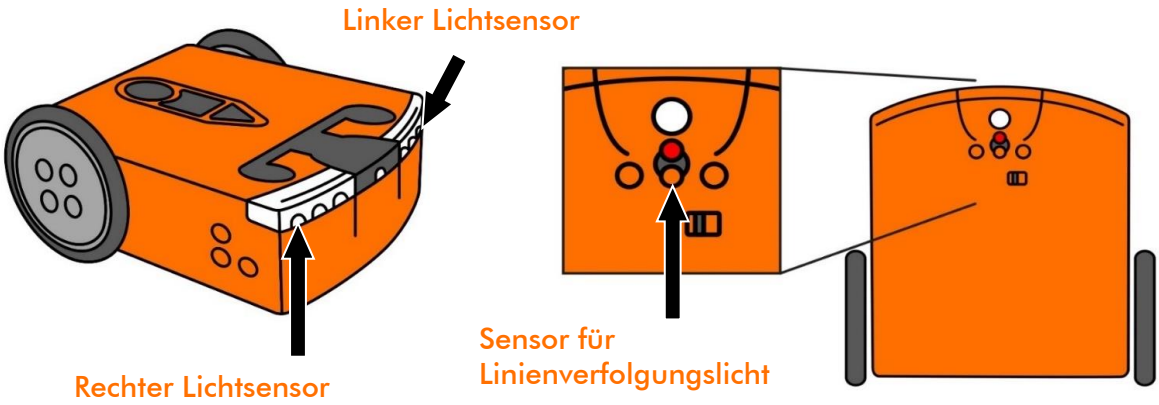
Name_____

U5-1.2: Edisons Lichtsensoren

Edison-Roboter haben verschiedene Sensoren, die verschiedene Dinge erkennen können. Eine Art von Sensoren, die Edison hat, sind die Lichtsensoren.

Aufgabe 1: Edisons Lichtsensoren

Edisons Lichtsensoren sind die Sensoren, mit denen Edison sichtbares Licht erkennen und messen kann. Edisons Hauptlichtsensoren befinden sich auf der Oberseite des Roboters, einer rechts und einer links vom Roboter. Edison hat auch einen dritten Lichtsensor, der sich unter dem Roboter befindet und als Teil des Linienverfolgungssensors arbeitet. Schau dir deinen Edison-Roboter an. Siehst du die verschiedenen Lichtsensoren?



Ähnlich wie der Infrarotempfänger von Edison infrarotes Licht erkennen kann, können die Lichtsensoren dazu benutzt werden, sichtbares Licht zu erkennen, das ist der Teil des Lichtspektrums, den die Menschen sehen können. Alle Sensoren für sichtbares Licht von Edison funktionieren im Grunde genommen auf die gleiche Weise wie der Lichtsensor in Edisons Linienverfolger, wenn man ihn benutzt, um reflektierende und nicht reflektierende Oberflächen unter dem Roboter zu erkennen.

Nicht vergessen

Der Linienverfolgungssensor funktioniert, indem er Licht von der roten LED im Linienverfolger auf die Oberfläche unter dem Roboter strahlt. Der Lichtsensor im Linienverfolgungsgerät misst dann, wie viel von diesem Licht von der Oberfläche aufprallt. Edison speichert den Wert des reflektierten Lichts als Lichtmesswert. Je mehr Licht zu Edison zurückreflektiert wird, desto höher ist der Lichtwert.

Name_____

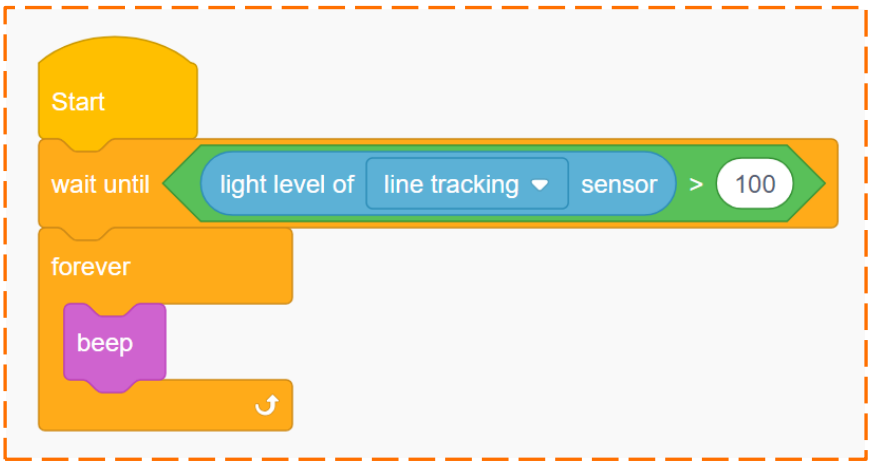
Edisons Lichtsensoren müssen sich nicht nur auf das reflektierte Licht von Edisons LEDs verlassen. Wir können diese Sensoren auch benutzen, um das sichtbare Licht zu erkennen, das von jeder Quelle in der Nähe von Edison kommt. Die Sensoren messen das erkannte Licht und speichern den Wert als Lichtmesswert. Je mehr Licht detektiert wird, desto höher ist der Lichtwert.

Wir können den Lichtmesswert von einem der Lichtsensoren als Wert in einem EdScratch-Programm verwenden.

Aufgabe 1: Licht-Alarm

Lass uns einen von Edisons Lichtsensoren benutzen, um einen Alarm auszulösen, der ertönt, wenn Edison genug Licht entdeckt.

Schau dir dieses EdScratch-Programm an:



Dieses Programm sagt Edison, dass er warten soll, bis der Lichtpegel des Linienverfolgungssensors größer als 100 ist, dann piepst er für immer.

Woran liegt das?

Die Lichtsensoren messen das erkannte sichtbare Licht und speichern den Wert als Lichtmesswert. Wie alle Sensordaten speichert Edison diesen Wert als Zahl. Der Ausdruck in diesem Programm sagt Edison, dass er den Wert des Lichtmesswertes des Linienverfolgungssensors mit der Zahl 100 vergleichen soll.

Warum 100? Denke daran, je mehr Licht erkannt wird, desto höher ist der Lichtwert. Edisons Lichtsensoren können Lichtmesswerte mit Werten von 0 bis knapp über 1000 liefern. Das macht 100 zu einem guten Startpunkt, um mit diesem Programm zu testen.

Schreibe das Lichtalarmprogramm und lade es auf deinen Edison-Roboter herunter. Bevor du die Play-Taste (Dreieck) drückst, um das Programm zu starten, bedecke den Sensor mit etwas wie deinem Daumen. Stelle sicher, dass der Lichtsensor komplett abgedeckt ist! Halte den Roboter so, dass der Line Tracking Lichtsensor von hellen Lichtern weg zeigt, wie z.B. Deckenlicht oder Sonnenschein, der von einem Fenster einfällt.

Sobald du den Roboter in Position gebracht hast und der Lichtsensor abgedeckt ist, drücke Play. Wenn du bereit bist, nimm den Daumen vom Lichtsensor weg und richte den Roboter auf eine Lichtquelle. Sobald der Roboter genug Licht entdeckt hat, wird gewartet, bis die Bedingung des Blocks erfüllt ist, und der Code geht zum nächsten Block weiter, wodurch der Pieptonblock 'Alarm' ausgelöst wird.

Aufgabe 2: Automatische Straßenlampe

Hast du jemals eine Straßenlampe gesehen, die angeht, wenn es draußen dunkel wird? Die Chancen stehen gut, dass dies mit einem Lichtsensor geschieht! Du kannst Edison dazu bringen, sich genauso zu verhalten, indem du die roten LEDs des Roboters einschaltest, wenn das Lichtniveau zu niedrig wird, und sie ausschaltest, wenn viel Licht vorhanden ist.

Schreibe ein Programm, das deinen Edison dazu bringt, sich wie eine lichtempfindliche Straßenlampe zu verhalten. Dein Programm sollte Edison dazu bringen, die roten LEDs einzuschalten, wenn der Roboter sehr wenig sichtbares Licht erkennt, aber die roten LEDs in der restlichen Zeit auszuschalten.

Tipp!

- Du brauchst nur einen der oberen Lichtsensoren von Edison zu benutzen, um eine Lichtstärkeanzeige für dieses Programm zu erhalten. Wähle entweder den linken oder den rechten Lichtsensor.
- Teste verschiedene Werte, mit denen du deine Lichtsensorwerte vergleichen kannst, um zu sehen, was am besten funktioniert.
- Fühlst du dich festgefahren? Betrachte das Programm aus Aufgabe 1. Wie kannst du es modifizieren, um dein automatisches Straßenlampenprogramm zu erstellen?

Lade dein Programm herunter und teste es in deinem Roboter. Setze Edison an eine Stelle mit viel Licht, dann an eine Stelle ohne viel Licht. Bringt dein Programm Edison dazu, wie eine automatische Straßenlampe zu funktionieren?

Name_____

1. Wie sieht dein automatisches Straßenlampenprogramm aus? Schreibe es auf.

Name_____

U5-1.2a: Edison die Motte

Hast du jemals bemerkt, wie manche fliegende Insekten von hellen Lichtern angezogen werden? Diese Art von Verhalten wird positiver Fototropismus genannt. Motten zeigen positiven Fototropismus, weshalb sie nachts um ein helles Licht herumschwärmen. Diese Art von Verhalten findet man auch bei Pflanzen, die zur Sonne hin wachsen.

Wir können Edison dazu bringen, dieses Verhalten zu imitieren, indem er dem hellsten Licht folgt, das der Roboter erkennt.

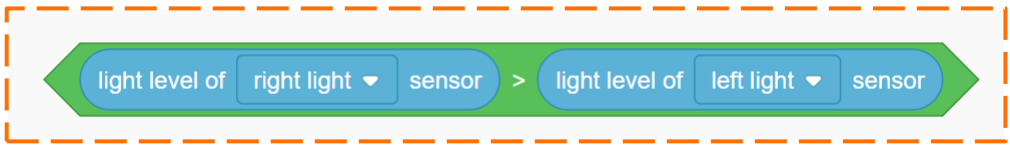
Was zu tun ist

Lasst uns Edison so programmieren, dass er sich wie eine Motte verhält, die einem hellen Licht folgt. Dein Programm sollte Edison dazu bringen, sich auf das hellste Licht zuzubewegen, das es entdeckt. Um dieses Programm zu schreiben, musst du die Lichtniveau-Messwerte der beiden obersten Lichtsensoren von Edison verwenden.

Nicht vergessen

Die Lichtsensoren messen das erkannte sichtbare Licht und speichern den Wert als Lichtmesswert. Wie alle Sensordaten speichert Edison diesen Wert als Zahl. Du kannst diesen Wert mit einem anderen Wert in einem EdScratch-Programm vergleichen und Edison sagen, wie er je nach Ergebnis reagieren soll.

Du kannst den Lichtniveau-Messwert von einem von Edisons Lichtsensoren mit einer festen Zahl vergleichen. Du kannst auch die Lichtniveau-Messwerte eines Lichtsensors mit den Lichtniveau-Messwerten eines anderen Lichtsensors vergleichen:



Ihr müsst beide oberen Lichtsensoren von Edison in eurem Programm verwenden. Immer wenn der richtige Lichtsensor den höheren Wert anzeigt, sollte sich der Roboter nach rechts bewegen. Immer wenn der linke Lichtsensor einen höheren Wert anzeigt, sollte sich der Roboter in diese Richtung bewegen.

Schreibe dein Programm in EdScratch, lade es dann herunter und teste es mit deinem Edison-Roboter. Benutze eine helle Lichtquelle, wie eine Taschenlampe, und teste, ob Edison dem Licht in der Umgebung folgt.

Name_____

Tipp!

Wenn der Raum, in dem du testest, sehr hell ist, wird Edison das Licht deiner Taschenlampe nicht erkennen können. Wenn es in dem Raum eine helle Lichtquelle gibt, wie z.B. viel Sonnenschein, der aus einem Fenster einfällt, kann dies deine Lichtquelle überstrahlen und der Roboter könnte stattdessen auf die andere Lichtquelle zusteuern!

Name_____

U5-1.2b: Edison, die Kakerlake

Einige Pflanzen und Tiere zeigen ein Verhalten, das als positiver Fototropismus bekannt ist. Diese Kreaturen werden vom Licht angezogen, wie Motten, die nachts um ein helles Licht schwärmen. Andere Kreaturen, wie einige Kakerlaken, meiden Licht. Dieses Verhalten wird als negativer Fototropismus bezeichnet.

Kannst du Edison dazu bringen, dieses Verhalten nachzuahmen und das Licht zu meiden?

Was ist zu tun?

Programmiere Edison so, dass er sich wie eine Kakerlake verhält, die sich bewegt und versucht, jegliches sichtbare Licht zu vermeiden. Als erstes solltest du dein Programm mit Pseudocode planen.

1. Schreibe deinen Pseudocode auf.

_____

_____

_____

_____

_____

_____

_____

Tipp!

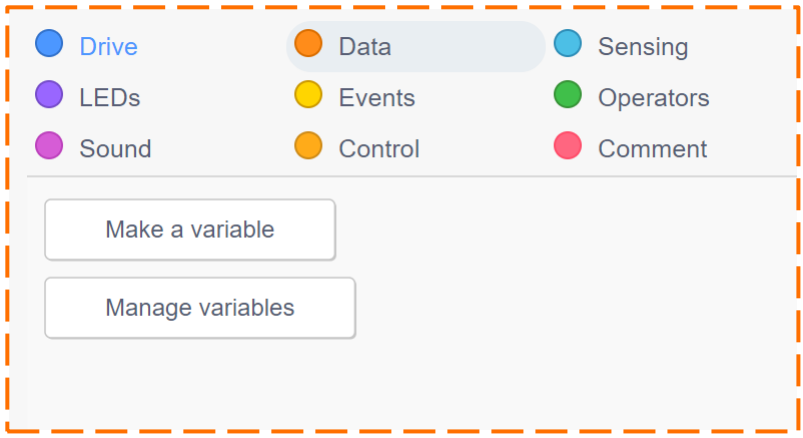
Du kannst den Lichtpegel, den einer von Edisons Lichtsensoren anzeigt, mit einer festen Zahl vergleichen. Du kannst auch die Lichtniveau-Messwerte von einem Lichtsensor mit den Lichtniveau-Messwerten eines anderen Lichtsensors vergleichen. Welcher Ansatz funktioniert am besten für das, was du in diesem Programm versuchst zu tun?

Schreibe dein Programm in EdScratch und benutze deinen Pseudocode als Anleitung. Lade dein Programm herunter und teste es in deinem Edison-Roboter. Benutze eine helle Lichtquelle, wie z.B. eine Taschenlampe, und teste, ob Edison sich wie eine Kakerlake verhält, indem er dem Licht ausweicht.

Name_____

U5-1.3: Variablen

In EdScratch gibt es eine Kategorie von Blöcken, die sich sehr von den anderen Kategorien unterscheidet, wenn du sie zum ersten Mal öffnest: die Kategorie Data.



Wenn du das erste Mal auf die Data-Kategorie klickst, stehen dir keine Blöcke zur Verfügung, die du in einem EdScratch Programm verwenden kannst. Stattdessen gibt es zwei Schaltflächen: die "Variable erstellen"-Schaltfläche und die "Variablen verwalten"-Schaltfläche (Manage variables). Mit Hilfe der Kategorie Data können wir Variablen erstellen, die wir in unseren EdScratch-Programmen verwenden können.

Fachjargon

Eine Variable ist ein Stück Speicher, das dazu dient, einen Wert in einem Programm zu speichern. Man kann sich eine Variable wie einen Container vorstellen, in dem man einige Informationen auf eine Art und Weise speichern kann, die für einen Computer Sinn ergibt.

In der Computerprogrammierung verwenden wir oft die gleiche Information mehrmals in einem einzigen Programm. Variablen machen es viel einfacher, dies zu tun. Durch die Verwendung von Variablen in Programmen können wir dem Computer sagen, dass er ein bestimmtes Stück Information in einer Variable speichern soll. Wir können diese Variable dann an verschiedenen Stellen im Programm verwenden. Jedes Mal, wenn der Computer die Variable sieht, ruft er die Informationen ab, die in der Variable gespeichert sind.

Wenn du in EdScratch auf die Schaltfläche Make a variable klickst, öffnet sich ein Popup-Fenster, in dem du aufgefordert wirst, deiner Variable einen Namen zu geben. Variablen

einen guten Namen zu geben, ist wichtig: du willst, dass der Name deiner Variable für dich Sinn macht und dir sagt, welche Informationen darin gespeichert sind.

Deine EdScratch-Variablennamen dürfen nur englische Kleinbuchstaben, englische Großbuchstaben, Zahlen und Unterstriche (_) enthalten. Andere Symbole, wie Ausrufezeichen (!) oder Leerzeichen, sind nicht erlaubt.

Woran liegt das?

Variablennamen müssen für dich und den Computer einen Sinn ergeben. Computersprachen haben oft Regeln darüber, welche Zeichen in Variablennamen verwendet werden können. Der Computer kann nur Variablen mit Namen verstehen, die diesen Regeln folgen.

Gib deiner Variable einen Namen, der dir hilft zu erkennen, welche Art von Information in der Variable gespeichert werden soll. Wenn du den Namen einer Variable ändern möchtest, nachdem du sie erstellt hast, kannst du dies über die Schaltfläche Manage variables tun.

Sobald du eine Variable erstellt und benannt hast, erscheint sie in der Kategorie "Data" zusammen mit einigen anderen speziellen Blöcken wie dem set-Block, dem Increment-Block und dem Dekrement-Block. Diese Blöcke kannst du dann zusammen mit deiner Variable in einem EdScratch-Programm verwenden.

Fachjargon

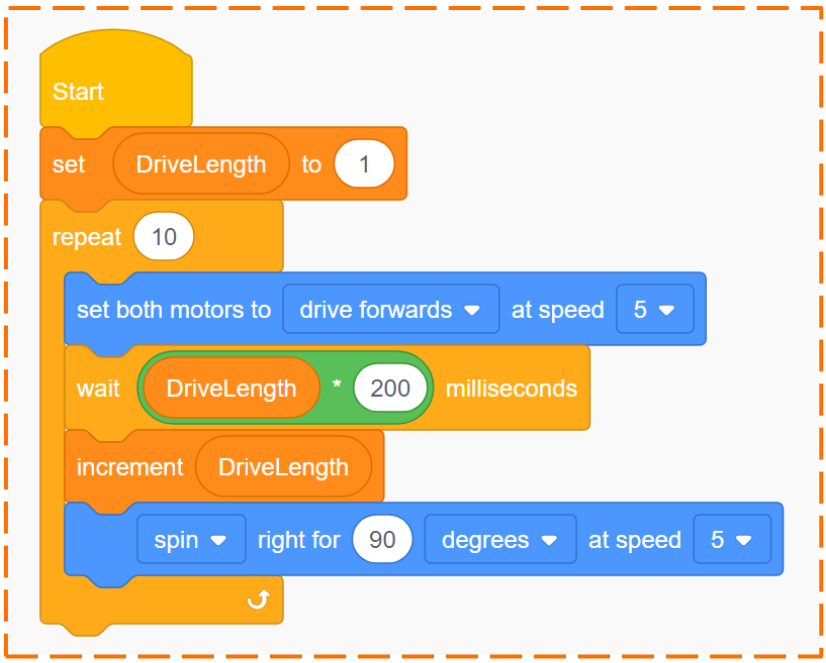
In der Computerprogrammierung bedeutet inkrementieren, um 1 zu erhöhen und dekrementieren, um 1 zu

Name_____

Aufgabe 1: Den Code zurückverfolgen

In EdScratch können wir Variablen verwenden, um verschiedene Werte zu speichern. Weil diese Werte Zahlen sind, können wir dann verschiedene Dinge mit diesen Zahlen machen, wie sie mit anderen Werten vergleichen oder Berechnungen mit ihnen machen.

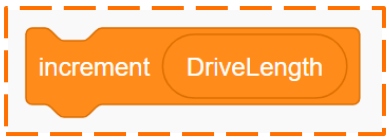
Schau dir das folgende Programm an:



Dieses Programm benutzt eine Variable namens DriveLength.

Es ist wichtig zu beachten, dass eine Variable einen Wert repräsentiert, der irgendwo in deinem Programm gesetzt ist. Deshalb musst du in deinem Programm immer Code verwenden, um dem Computer mitzuteilen, wie die Variable gesetzt werden soll.

Genau das macht der set-Block in diesem Programm. Am Anfang des Programms setzt der set-Block die Variable DriveLength auf 1. Der Wert von DriveLength wird jedoch nicht für immer auf 1 gesetzt bleiben. Das liegt daran, dass dieses Programm einen anderen Block aus der Data-Kategorie verwendet, den increment-Block:



Der increment-Block befindet sich innerhalb der Wiederholungsschleife (repeat) und ändert die Variable DriveLength auf einen neuen Wert. Was ist dieser Block, der die Variable DriveLength auf einen neuen Wert ändert?

Das hängt davon ab, wo im Programm du dich gerade befindest. Mit anderen Worten, es hängt davon ab, wie oft die Wiederholungsschleife durchlaufen wurde.

Name_____

Woran liegt das?

Einer der Hauptgründe, warum wir Variablen in Programmen verwenden, ist, dass eine Variable eine Information enthalten kann, auch wenn sich der Wert dieser Information ändert. Das mag wirklich verwirrend klingen, aber denk mal so darüber nach:

Nehmen wir an, du hast eine Variable namens YearsOld und die Information, die diese Variable enthält, ist dein Alter. An deinem ersten Geburtstag wurde YearsOld auf 1 gesetzt. Danach wird YearsOld jedes Jahr an deinem Geburtstag inkrementiert, was es auf einen neuen Wert ändert: (YearsOld + 1).

Ein Jahr nach deinem ersten Geburtstag wird YearsOld hochgezählt. Da der Wert von YearsOld 1 war, und da 1 + 1 = 2, wurde der neue Wert von YearsOld 2. Dies wiederholt sich an deinem nächsten Geburtstag, wobei der Wert von YearsOld erhöht wird und 3 wird.

Der Wert von YearsOld ändert sich an jedem Geburtstag, aber die Art der Information ändert sich nicht. YearsOld ist dein Alter, egal wie viele Geburtstage du hast!

Wenn unser Programm zum ersten Mal startet, wird DriveLength auf 1 gesetzt, aber weil es einen increment-Block innerhalb des Schleifencodes gibt, wird der Wert jede Schleife ändern.

DriveLength wird auch an einer anderen Stelle innerhalb des geloopten Codes in diesem Programm verwendet:

Der Eingabewert dieses Warteblocks hängt von dem Wert von DriveLength ab und ändert ebenfalls jede Schleife.

Sei aber vorsichtig!

Während der Wert des wait-Blocks vom Wert von DriveLength abhängt, wird dieser Warteblock den Wert von DriveLength nicht ändern. Nur ein Block aus der Kategorie Data kann den Wert einer Variablen ändern.

Vielleicht fällt dir auch auf, dass dieser wait-Block auf Millisekunden und nicht auf Sekunden eingestellt ist.

Name_____

Woran liegt das?

Immer wenn du eine Variable oder einen Block aus der Kategorie Operators als Input in einem Warteblock verwenden möchtest, musst du diesen speziellen Millisekunden-Warteblock (millisecond wait) verwenden. Edison 'denkt' eigentlich in Millisekunden, nicht in Sekunden, also ermöglicht die Verwendung dieses Blocks Berechnungen, die Edison verstehen kann.

Denke daran, 1 Sekunde = 1000 Millisekunden.

Lass uns das Programm zurückverfolgen, um herauszufinden, wie der Wert der Variablen DriveLength und der Eingabewert des Warteblocks an verschiedenen Stellen sein werden, wenn der Code in diesem Programm läuft.

Nicht vergessen

Code zurückverfolgen bedeutet, ein Programm Zeile für Zeile abzuarbeiten und wichtige Werte aufzuzeichnen.

1. Gehen wir das Programm durch, um die Werte herauszufinden. Füllt die Tabelle aus, wie der Startwert der Variablen DriveLength am Anfang jeder Schleifenwiederholung sein wird, wie der Eingabewert des wait Block in dieser Schleife sein wird, und wie der neue Wert von DriveLength nach dem increment Block innerhalb der Schleife sein wird. Die ersten beiden Zeilen sind bereits für dich ausgefüllt.

In loop #	Anfangswert von DriveLength	Warteblock-Eingangswert (in Millisekunden)	Neuer Wert von DriveLength
1	1	200	2
2	2	400	3
3			
5			
7			
10			

Name_____

Aufgabe 2: Schreibe und führe das Programm aus

Was wird das Programm den Roboter tun lassen, wenn du ihn in Edison laufen lässt?

Schreibe das Programm in EdScratch. Lade es herunter und lasse es in deinem Edison-Roboter laufen, um zu sehen, was es macht. Dann beantworte die Fragen.

2. Was macht Edison, wenn du dieses Programm ausführst? In welcher 'Form' fährt der Roboter?

_____

_____

3. Schau dir den Code im Programm an. Erkläre, warum der Roboter in dem Muster fährt, das du siehst, wenn du das Programm in Edison ausführst. Was in dem Code bringt den Roboter dazu, sich in diesem Muster zu bewegen?

_____

_____

_____

_____

Name_____

U5-1.3a: Spiralförmige Spinnenfalle

Manche Spinnen bauen Netze, die sich spiralförmig zu einem zentralen Punkt in der Mitte winden. Kannst du Edison so programmieren, dass er so fährt, dass der Roboter sich nach innen dreht, wie eine Spinne, die eine Falle legt?

Was zu tun ist

Schreibe ein Programm in EdScratch, das Edison dazu bringt, in einer nach innen gedrehten Form zu fahren. Edison soll fahren, dann wenden, dann immer wieder wiederholen und sich nach innen spiralförmig drehen.

Dein Programm sollte eine Variable verwenden, die dir hilft zu kontrollieren, wie weit Edison jedes Mal fährt.

Tipp!

Du solltest dir das Programm in Aktivität U5-1.3 betrachten, um dir Inspiration für dein Programm zu holen.

Denke darüber nach, wie weit Edison jeden Abschnitt im Vergleich zum vorherigen Abschnitt fahren soll. Du wirst auch entscheiden müssen, wie weit Edison zwischen den einzelnen Fahrabschnitten abbiegen soll.

Lade dein Programm herunter und teste es mit deinem Edison-Roboter. Experimentiere mit verschiedenen Eingabewerten, um zu sehen, was am besten funktioniert.

Mini-Herausforderung!

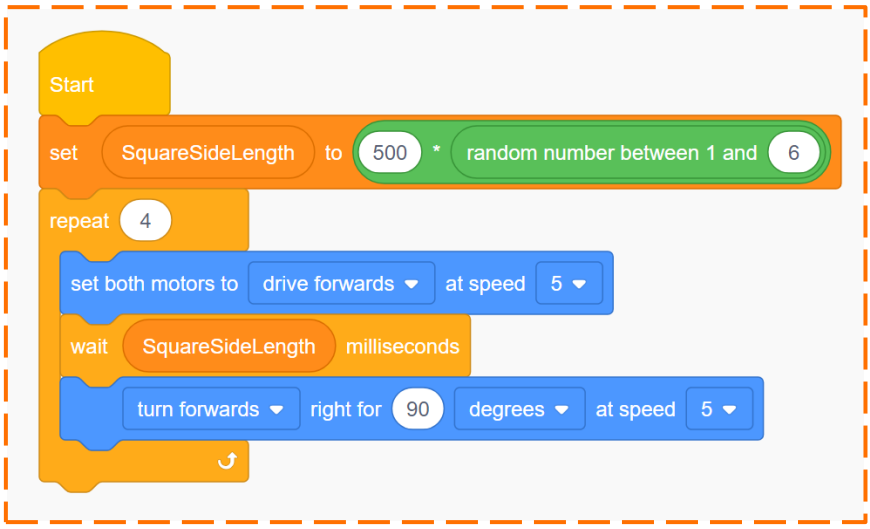
Kannst du Edison dazu bringen, eine 'klebrige Spinnenfadenspur' hinter dem Roboter zu hinterlassen, während er fährt? Benutze etwas, zum Beispiel eine Schnur, um eine Spinnenseide darzustellen. Versuche, eine Lösung zu finden, damit Edison beim Fahren eine Spur hinterlässt.

Name_____

U5-1.3b: Fahre ein zufälliges Quadrat

Mit Hilfe von Variablen können wir Programme erstellen, die viele interessante Dinge tun können.

Schau dir dieses Programm an:



Wenn Edison dieses Programm ausführt, wird der Roboter in einem Quadrat fahren. Aber wie groß wird dieses Quadrat sein?

Was zu tun ist

Schreibe das Programm in EdScratch, lade es dann herunter und starte es mit deinem Edison-Roboter. Sieh zu, wie Edison das Quadrat fährt. Nun starte das Programm erneut. Edison wird wieder in einem Quadrat fahren, aber die Chancen stehen gut, dass dieses Quadrat eine andere Größe haben wird als das erste!

Wenn du den Zufallszahlenblock (random number) innerhalb des set Blocks so verwendest, bedeutet das, dass sich der Wert der Variable bei jedem Programmstart ändert.

1. Jedes Mal, wenn das Programm läuft, wird die Variable SquareSideLength auf einen von 6 verschiedenen Werten gesetzt. Füllt die Tabelle mit den verschiedenen Werten aus, auf die SquareSideLength gesetzt wird, je nachdem, welche Zufallszahl gewählt wird.

Name_____

Wenn die Zufallszahl ist:	Wert SquareSideLength
1	
2	
3	
4	
5	
6	

Mini-Herausforderung!

Wie kann man erkennen, welche Zufallszahl im Programm verwendet wurde? Überlegt euch, wie ihr das Zufallsquadrat-Programm so modifizieren könnt, dass der Roboter ein zufälliges Quadrat fährt, aber auch irgendwie signalisiert, welcher Wert in SquareSideLength verwendet wurde. Modifiziere das Programm und teste es in deinem Edison-Roboter aus. Hat es funktioniert?

Name_____

U5-1.4 Die Verwendung von Variablen mit Sensordaten

Eine der interessantesten Arten, wie wir Variablen in EdScratch verwenden können, ist die Speicherung und Verwendung von Daten von Edisons Sensoren. Da alle Sensoren von Edison Daten als Werte an Edison zurücksenden, kannst du einen Wert von einem Sensor in einer Variable speichern. Du kannst auch Sensordaten verwenden, um eine Variable zu beeinflussen, z.B. indem du die Variable jedes Mal inkrementierst, wenn ein Sensor etwas feststellt. Indem wir Variablen und Sensoren zusammen in einem Programm verwenden, können wir Edison dazu bringen, auf Sensordaten auf unterschiedliche Weise zu reagieren.

Nicht vergessen

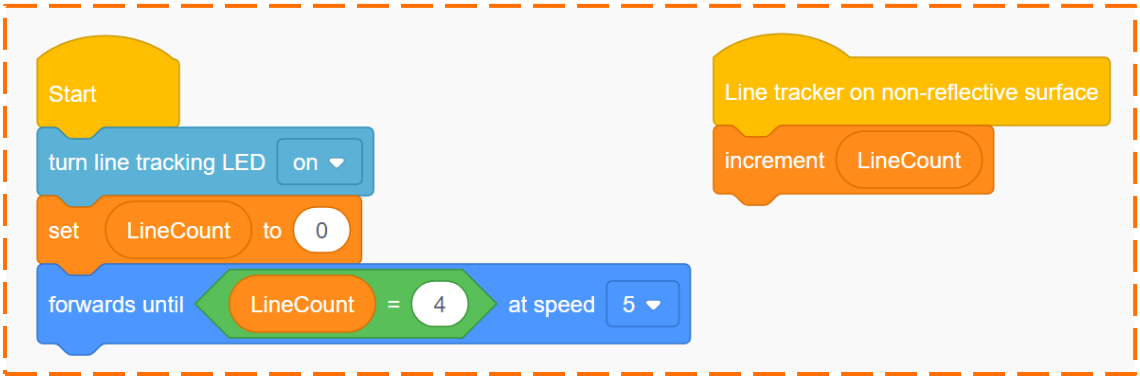
Eine Variable ist ein Stück Speicher, das verwendet wird, um einige Informationen in einem Programm zu speichern.

Versuchen wir, eine Variable mit Edisons Linienverfolgungssensor zu verwenden, um Edison so lange fahren zu lassen, bis der Roboter vier schwarze Linien entdeckt.

Was zu tun ist

Mit Edisons Linienverfolgungssensor kannst du ein Programm erstellen, das Edison sagt, dass er fahren soll, bis er eine schwarze Linie entdeckt. Bei dieser Aktivität musst du den Roboter dazu bringen, über mehrere schwarze Linien zu fahren und erst anzuhalten, wenn er die vierte schwarze Linie entdeckt hat.

Schau dir das folgende Programm an:



1. Was geht in diesem Programm vor? Wenn dieses Programm nach Edison heruntergeladen wird, was wird es dem Roboter sagen, was er tun soll?

_____

_____

_____

_____

Name_____

2. Dieses Programm hat eine Variable namens LineCount in sich. Warum glaubst du, hat der Programmierer die Variable so genannt?

_____

_____

Probier es aus

Schreibe das Programm in EdScratch und lade es dann auf deinen Edison-Roboter herunter. Teste das Programm mit dem Arbeitsblatt U5-1 oder mache deinen eigenen Testraum, indem du vier parallele schwarze Linien auf einer weißen Fläche, wie z.B. einem großen Poster, Tisch oder dem Boden, zeichnest. Du kannst die Linien so weit voneinander entfernt anbringen, wie du willst.

3. Wenn wir einen von Edisons Sensoren mit einer Variablen verwenden, können wir alle möglichen Programme erstellen, die Edison auf verschiedene Weise nutzen. Wie könnte man Edison noch dazu bringen, mit Variablen und Sensordaten zu arbeiten? Lass dir mindestens eine Idee für ein Programm einfallen, das eine Variable in Kombination mit Sensordaten verwendet. Beschreibe deine Idee.

_____

_____

_____

_____

_____

4. Kann deine Idee kodiert werden? Versuche, dein Programm in EdScratch zu schreiben. Wie ist es gelaufen? Warst du in der Lage, dein Programm erfolgreich zu schreiben und es mit Edison zu testen? Wenn ja, hat es funktioniert? Schreibe über den Prozess, den du genommen hast, um deine Idee in ein Programm zu verwandeln.

_____

_____

_____

_____

_____

_____

_____

_____

_____

Name_____

U5-1.4a: Edison der Sprinter

Der 400-Meter-Lauf ist ein Sprint-Event in vielen Leichtathletik-Wettbewerben, einschließlich der Olympischen Sommerspiele. Dieser Wettkampf ist einer der längsten 'Sprint'-Wettkämpfe und das unterscheidet ihn von anderen, kürzeren Wettkämpfen. Die meisten Athleten, die die 400-Meter-Strecke laufen, wissen, dass sie nicht mit 100%iger Anstrengung vom Start bis zum Ziel sprinten können. Stattdessen teilen sie das Rennen in verschiedene Phasen auf, um Ausdauer und Geschwindigkeit auszugleichen.

Wir können Edison so programmieren, dass er wie ein 400-Meter-Spezialist arbeitet, indem er die Geschwindigkeit ändert, während der Roboter eine Strecke durchläuft.

Was zu tun ist

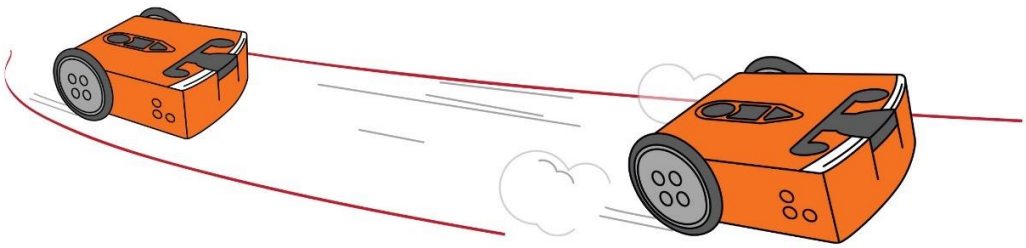
Schreibe ein Programm, das Edison dazu bringt, sich wie ein 400-Meter-Spezialist zu verhalten und die Geschwindigkeit zu erhöhen, wenn der Roboter Markierungen (schwarze Linien) auf einem Kurs passiert. Dein Programm sollte Edisons Linienverfolgungssensor und eine Variable verwenden, um zu zählen, wie viele Linien der Roboter angetroffen hat.

Tipp!

Du solltest dir das Programm in Aktivität U5-1.4 betrachten, um dir Inspiration für dein Programm zu holen.

Teste dein Programm mit dem Arbeitsblatt U5-1. Bei dieser Aktivität ist die vierte Linie die Ziellinie, also musst du sicherstellen, dass der Roboter diese Linie überquert und nicht einfach stehen bleibt, sobald er diese Linie entdeckt hat.

Du kannst auch deinen eigenen Testraum gestalten, indem du vier parallele schwarze Linien auf einer weißen Fläche, z.B. einem großen Poster, einem Tisch oder dem Boden, markierst. Du kannst die Linien so weit voneinander entfernt anbringen, wie du willst.



Name_____

U5-1.4b: Edison-kontrollierte Flaggenmaschine

Edisons Infrarot (IR)-Lichtsensor ist der Sensor, der Edison Infrarotlicht ausstrahlen und erkennen lässt. Wir können den IR-Sensor benutzen, um Nachrichten an andere Edison-Roboter zu senden oder Nachrichten von ihnen zu empfangen. Ein Roboter kann mit seinen beiden IR-LEDs eine Infrarot-Nachricht aussenden, die der IR-Empfänger eines anderen Roboters erkennen kann.

Nicht vergessen

Infrarot wird manchmal als IR abgekürzt. In EdScratch sind IR-Meldungsblöcke Blöcke, die sich auf Edisons Infrarot-Meldungen mit Hilfe der Infrarot-LEDs und des Infrarot-Empfängers beziehen.

Um eine Infrarot-Nachricht mit deinem Edison-Roboter in EdScratch zu versenden, musst du den Sende-IR-Nachrichtenblock verwenden, der einen Eingabeparameter hat, der es dir erlaubt, eine bestimmte Nachricht zu senden. Du kannst die Nachricht ändern, indem du den Wert des Eingabeparameters im Sende-IR-Nachrichtenblock änderst. Der Sende-IR-Meldungsblock hat einen Eingabebereich von 0 bis 255. Mit anderen Worten, Edison kann 256 verschiedene 'Nachrichten' senden und empfangen.

Indem wir IR-Nachrichten mit unterschiedlichen Werten verwenden, können wir ein Programm erstellen, das Edison anweist, auf eine Weise zu reagieren, wenn der Roboter eine IR-Nachricht erkennt, und auf eine andere Weise, wenn er eine andere Nachricht erkennt. Damit das funktioniert, müssen wir eine Variable verwenden, die die Daten speichert, die Edison mit seinem IR-Sensor empfängt.

Lasst uns versuchen, eine Flaggenmaschine zu bauen und zu programmieren, indem wir Edisons Motorausgänge benutzen, die ihr mit Nachrichten von einem anderen Edison-Roboter steuern könnt.

Nicht vergessen

Eine Variable ist ein Stück Speicher, das verwendet wird, um einige Informationen in einem Programm zu speichern. Da alle Sensoren von Edison Daten als Werte an Edison zurücksenden, kannst du diesen Wert in einer Variablen speichern.

Was zu tun ist

Das Ziel dieser Aktivität ist es, eine Flaggenmaschine zu bauen und zu programmieren, mit der du dazu beitragen kannst, dass das Lernen für einen Test mehr Spaß macht. Für diese Aktivität müsst ihr mit einem Partner zusammenarbeiten und zwei Edison-Roboter benutzen. Ein Roboter wird die Flaggenmaschine sein, die eine zweiseitige Flagge mit dem Wort 'Ja' auf der einen Seite der Flagge und dem Wort 'Nein' auf der anderen Seite

haben wird. Diese 'flag machine'-Roboter wird Nachrichten empfangen. Der andere Roboter wird der Controller-Bot sein, der die Nachrichten verschickt.

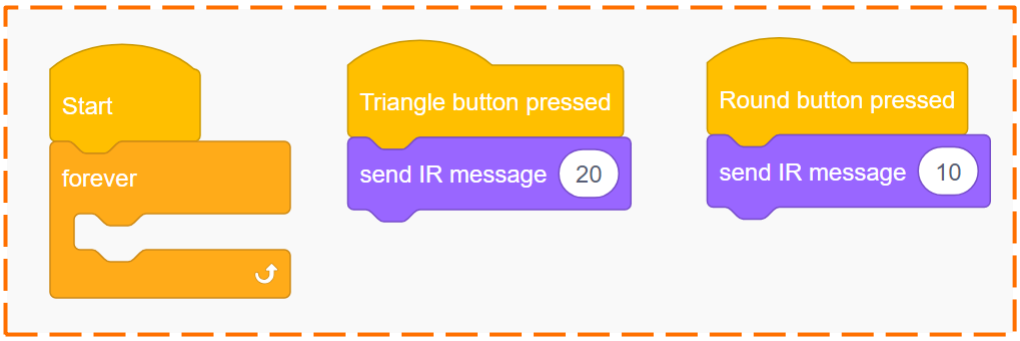
Dieses Projekt besteht aus drei Teilen: Entwurf und Bau der Flaggenmaschine, Programmierung der Flaggenmaschine und Programmierung des Controller-Bots. Wenn eure Flaggenmaschine fertig ist und beide Roboter programmiert sind, könnt ihr mit eurem Partner zusammenarbeiten, indem ihr euch abwechselnd gegenseitig abfragt. Du kannst deinen Controller-Bot benutzen, um die Flaggenmaschine zu bewegen und so signalisieren, ob dein Partner deine Frage richtig beantwortet hat oder nicht!

Als erstes musst du deinen Controller-Bot programmieren.

Das Controller-Bot-Programm

Der Edison-Roboter, der die IR-Nachrichten aussendet, ist der Controller-Bot. Dieser Roboter muss in der Lage sein, zwei verschiedene Nachrichten zu verschicken. Eine Nachricht wird die Flaggenmaschine anweisen, die 'Ja'-Seite der Flagge zu zeigen. Die andere Nachricht sagt der Flaggenmaschine, dass sie die 'Nein'-Seite der Flagge zeigen soll.

Schau dir dieses Programm an:



Du kannst es als dein Controller-Bot-Programm benutzen. Dieses Programm sagt dem Edison, der als Controller-Bot fungiert, dass er eine IR-Nachricht sendet, sobald du den runden oder dreieckigen Knopf auf dem Controller-Bot drückst. Wenn du auf die Dreieckstaste drückst, sendet der Controller-Bot eine IR-Nachricht 20 aus. Wenn du jedoch den runden Knopf drückst, sendet der Controller-Bot stattdessen die IR-Nachricht 10 aus.

In deinem Controller-Bot-Programm kannst du für deine beiden IR-Meldungen beliebige Werte zwischen 0 und 255 verwenden. Die beiden Nachrichten müssen jedoch unterschiedliche Werte haben, damit der empfangende Roboter die Nachrichten unterscheiden kann.

Die Werte, die ihr mit eurem Controller-Bot sendet, müsst ihr auch in eurem Flag-Machine-Programm verwenden.

1. Nur wenn du planst, wie deine beiden Edison-Roboter zusammenarbeiten werden, kann deine Kreation funktionieren. Arbeite mit deinem Partner zusammen, um zu

entscheiden, welcher Knopf auf dem Controller-Bot was tun wird. Schreibe deinen Plan auf.

Runder Knopf:

Wenn du den runden Knopf auf dem Controller-Bot drückst, sendet er die IR-Meldung _____ und das wird der Flaggenmaschine sagen, dass sie die _____ Seite der Flagge zeigen soll.

Dreieck-Button:

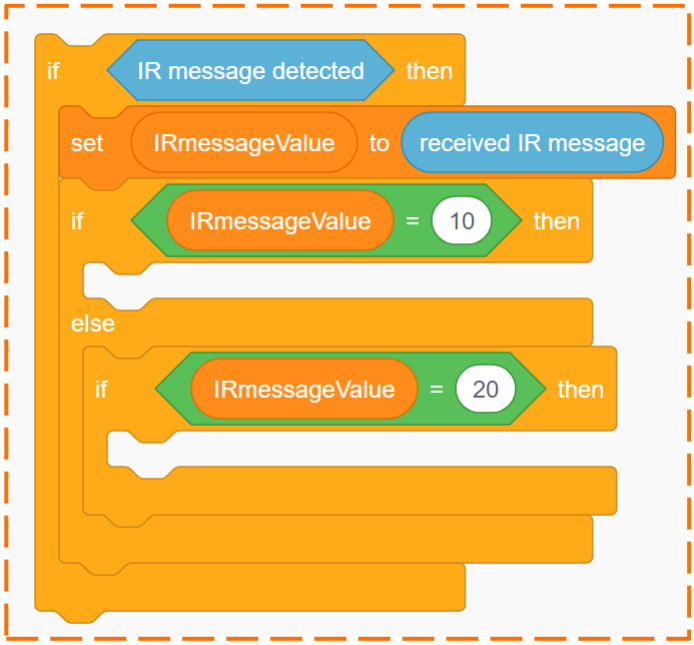
Wenn du die Dreieckstaste auf dem Controller-Bot drückst, sendet er die IR-Meldung _____ und das wird die Flaggenmaschine anweisen, die _____ Seite der Flagge zu zeigen.

Du wirst diesen Plan benutzen müssen, wenn du dein Flaggenmaschinenprogramm entwirfst.

Die Flaggenmaschine und das Flaggenmaschinenprogramm

Gestalte deine Flaggenmaschine so, dass sie Edisons Ausgänge benutzt, um die zweiseitige Flagge zu bewegen, wenn sie die verschiedenen IR-Meldungen empfängt. Die Flaggenmaschine sollte die 'Ja'-Seite der Flagge zeigen, wenn sie die eine IR-Meldung empfängt und die 'Nein'-Seite, wenn sie die andere IR-Meldung empfängt. Benutze den Plan, den du mit deinem Controller-Bot erstellt hast, um dir zu helfen.

Schau dir den folgenden Code an:



Dies ist kein vollständiges Programm, aber du kannst diesen Code verwenden, um dir beim Erstellen deines Flaggenmaschinen-Programms zu helfen.

Dieser Abschnitt des Codes verwendet eine Variable namens IRmessageValue. Der Wert dieser Variable ist so eingestellt, dass er dem Wert der empfangenen IR-Meldung entspricht. Wenn du mit Sensorwerten arbeitest, solltest du alle Werte, die du in deinem Programm verwenden möchtest, immer in einer Variablen speichern. Dann kannst du dein Programm diese Variable überall im Programm auf die Art und Weise verwenden lassen, die du brauchst.

Woran liegt das?

Computer verfolgen die Werte von Sensoren nicht, es sei denn, du sagst es ihnen. Wenn du einem Computer sagst, dass er einen Wert in einer Variablen speichern soll, sagst du ihm, dass er sich diesen Wert merken soll.

Erinnere dich daran, dass der IR-Detektor von Edison immer eingeschaltet ist und immer wieder nach Daten sucht. Der Roboter merkt sich nicht ewig jeden Wert, den er erkennt. Er behält den Wert nur so lange in seinem Arbeitsspeicher, bis er in einem Programm aufgerufen oder durch einen neuen Messwert ersetzt wird. Indem du den Wert der IR-Meldung in einer Variablen speicherst, sagst du Edison, dass er diesen Wert festhalten soll, damit du etwas damit machen kannst.

Der Wert bleibt so lange in der Variable, bis der Sensor eine neue IR-Meldung

Programmiere die Flaggenmaschine mit einer Variablen, um alle IR-Meldungen, die der Roboter erkennt, zu speichern. Wenn du deine Kreation gebaut und programmiert hast, probiere sie aus!

Name_____

U5-1.4c: Hey Edison, wo soll ich hin?

Edisons Infrarot (IR)-Lichtsensor ist der Sensor, der Edison Infrarotlicht aussenden und erkennen lässt. Wir können den IR-Sensor benutzen, um Nachrichten an andere Edison-Roboter zu senden oder von ihnen zu empfangen. Ein Roboter kann mit seinen beiden IR-LEDs eine Infrarot-Nachricht aussenden, die der IR-Empfänger eines anderen Roboters erkennen kann.

Nicht vergessen

Infrarot wird manchmal als IR abgekürzt. In EdScratch sind IR-Meldungsblöcke Blöcke, die sich auf Edison's Infrarot-Meldungen mit Hilfe der Infrarot-LEDs und des Infrarot-Empfängers beziehen.

Um eine Infrarot-Nachricht mit deinem Edison-Roboter in EdScratch zu versenden, musst du den Sende-IR-Nachrichtenblock verwenden, der einen Eingabeparameter hat, der es dir erlaubt, eine bestimmte Nachricht zu senden. Du kannst die Nachricht ändern, indem du den Wert des Eingabeparameters im Sende-IR-Nachrichtenblock änderst. Der Sende-IR-Meldungsblock hat einen Eingabebereich von 0 bis 255. Mit anderen Worten, Edison kann 256 verschiedene 'Nachrichten' senden und empfangen.

Indem wir IR-Nachrichten mit unterschiedlichen Werten verwenden, können wir ein Programm erstellen, das Edison anweist, auf verschiedene Weise zu reagieren, je nachdem, welche IR-Nachricht er erkennt. Damit das funktioniert, müssen wir eine Variable verwenden, die die Daten speichert, die Edison mit seinem IR-Sensor empfängt.

Versuchen wir zwei Edison-Roboter zu benutzen, einen fahrenden und einen navigierenden, um durch ein Schatzlabyrinth zu kommen. Die Roboter müssen verschiedene IR-Nachrichten verwenden, um miteinander zu kommunizieren, damit sie an den richtigen Ort gelangen.

Nicht vergessen

Eine Variable ist ein Stück Speicher, das verwendet wird, um einige Informationen in einem Programm zu speichern. Da alle Sensoren von Edison Daten als Werte an Edison zurücksenden, kannst du diesen Wert in einer Variablen speichern.

Name_____

Was zu tun ist

Für diese Aktivität benötigst du zwei Edison-Roboter: ein Roboter wird der Navigator-Roboter sein und der andere der Fahrer-Roboter. Der Fahrerroboter wird sich durch das Schatzlabyrinth bewegen, je nach den Nachrichten, die er vom Navigatorroboter erhält.

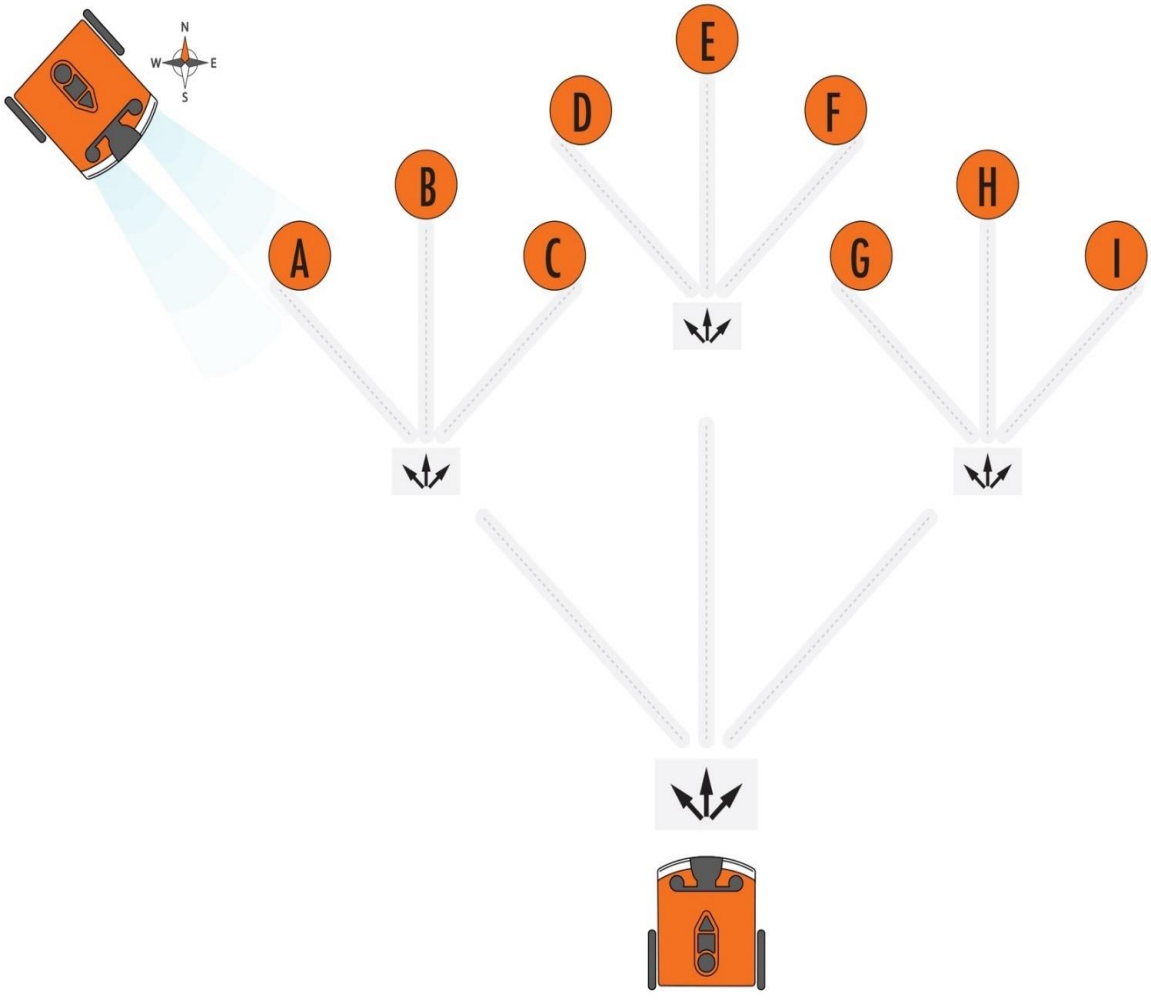
Das Projekt besteht aus drei Teilen: dem Erstellen des Schatzlabyrinths, dem Programmieren des Fahrerroboters und dem Programmieren des Navigatorroboters.

Als erstes müsst ihr euer Schatzlabyrinth erstellen, das ihr als Testraum benutzen werdet.

Das Schatzkarten-Labyrinth

Erstelle eine verzweigte Schatzkarte, die viele verschiedene mögliche Endpositionen hat. Euer Fahrer-Bot muss durch dieses Labyrinth navigieren, um zu der Stelle zu gelangen, an der sich der Schatz befindet. Deine Karte muss groß genug sein, damit Edison fahren kann, aber nicht so groß, dass die beiden Roboter sich keine Nachrichten hin und her schicken können. Du brauchst auch einen Platz, an dem der Navigator-Bot sitzen kann, um Befehle zu senden.

Die Grundform deines Testfeldes für die Schatzkarte muss in etwa so aussehen:



Alle Wege in eurer Karte sollten gleich lang sein. Wähle aus, wie lang die Pfade sein werden, wie viele Optionen an jeder Kreuzung herauskommen und wie viele Kreuzungen es vor den endgültigen Schatzplätzen gibt. Arbeitet zusammen, um euren Testplatz auf der Schatzkarte zu gestalten und zu dekorieren.

Tipp!

Benutze so etwas wie Buchstaben oder Zahlen, um die verschiedenen möglichen Schatzstellen zu markieren. Du weißt noch nicht, wo der echte Schatz sein wird!

Das Navigatorbot-Programm

Der Edison-Roboter, der sich nicht bewegt, sondern Anweisungen an den Fahrerroboter sendet, ist der Navigatorroboter. Du wirst nicht in der Lage sein, das endgültige Programm des Navigatorroboters zu schreiben, bis du weißt, wo auf der Schatzkarte du dem Fahrerroboter Anweisungen schickst.

Der Navigator-Bot wird dem Fahrer-Bot mit Hilfe einer Reihe von verschiedenen IR-Meldungen 'Befehle' senden. Jede dieser Nachrichten wird den Fahrer-Bot anweisen, eine andere Aktion auszuführen. Wie viele verschiedene Nachrichten euer Navigator-Bot senden muss, hängt von eurer Schatzkarte ab.

Wenn eure Karte zum Beispiel drei Pfade hat, die aus jeder Kreuzung herausführen, müsst ihr drei verschiedene IR-Nachrichten senden können: eine, die dem Fahrer-Bot sagt, dass er den linken Pfad nehmen soll, eine, die dem Fahrer-Bot sagt, dass er den mittleren Pfad nehmen soll, und eine, die dem Fahrer-Bot sagt, dass er den rechten Pfad nehmen soll.

Du brauchst noch eine weitere IR-Meldung. Diese Nachricht wird vom Fahrer-Bot an den Navigator-Bot gesendet, wenn der Fahrer-Bot bereit für den nächsten Befehl ist. Diese Nachricht wird die Art und Weise sein, wie der Fahrerroboter sagt: "Ich bin an der nächsten Kreuzung. Wohin soll ich jetzt gehen?"

Jede der von dir gesendeten IR-Nachrichten muss einen anderen Wert haben, damit der empfangende Roboter sie unterscheiden kann. Du kannst für deine IR-Meldungen beliebige Werte zwischen 0 und 255 verwenden. Achte darauf, dass deine Nachrichten sowohl in deinem Navigatorbot- als auch in deinem Driverbot-Programm übereinstimmen.

1. Wie viele Nachrichten wird euer Navigator-Bot brauchen, um dem Fahrer-Bot senden zu können? Welchen Wert wirst du für jede Nachricht verwenden (einschließlich der Nachricht des Treiber-Bots, um nach dem nächsten Befehl zu fragen)? Was wird der Treiber-Bot tun müssen, wenn er jede einzelne Nachricht

Name_____

vom Navigator-Bot erhält? Arbeitet zusammen und entwerft euren Plan. Macht euch Notizen, die euch beim Schreiben eurer EdScratch-Programme helfen sollen.

_____

_____

_____

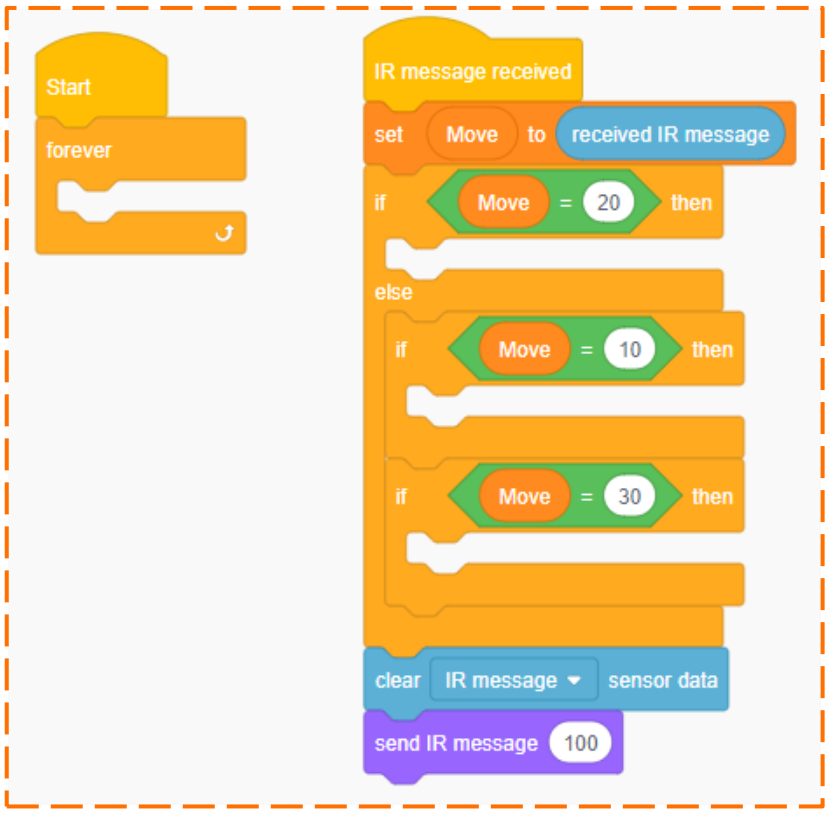
_____

Wert der IR-Meldung	Edison schicken (Navigator oder Fahrer)	Der Empfang von Edison wird...

Das Treiber-Bot-Programm

Der Edison-Roboter, der die IR-Nachrichten empfängt und nach diesen Befehlen fährt, ist der Fahrerroboter.

Schau dir den folgenden Code an:



Dies ist kein vollständiges Programm, aber du kannst diesen Code benutzen, um dein Treiber-Bot-Programm zu erstellen. Dieser Teil des Codes benutzt eine Variable namens Move. Der Wert dieser Variable ist so eingestellt, dass er dem Wert der empfangenen IR-Meldung entspricht. Wenn du mit Sensorwerten arbeitest, solltest du alle Werte, die du in deinem Programm verwenden möchtest, immer in einer Variable speichern. Dann kannst du dein Programm diese Variable überall im Programm auf die Art und Weise verwenden lassen, die du brauchst.

Name_____

Woran liegt das?

Computer verfolgen die Werte von Sensoren nicht, es sei denn, du sagst es ihnen. Wenn du einem Computer sagst, er soll einen Wert in einer Variablen speichern, sagst du ihm, er soll sich diesen Wert merken.

Erinnere dich daran, dass der IR-Detektor von Edison immer eingeschaltet ist und immer wieder nach Daten sucht. Der Roboter merkt sich nicht ewig jeden Wert, den er erkennt. Er behält den Wert nur so lange in seinem Arbeitsspeicher, bis er in einem Programm aufgerufen oder durch einen neuen Messwert ersetzt wird. Indem du den Wert der IR-Meldung in einer Variablen speicherst, sagst du Edison, dass er diesen Wert festhalten soll, damit du etwas damit machen kannst.

Der Wert bleibt so lange in der Variable, bis der Sensor eine neue IR-Meldung erkennt und der eingestellte Block den alten Wert durch den neuen ersetzt.

Dein Treiber-Bot-Programm muss nach einer IR-Meldung suchen und dann den Wert dieser Meldung als Variable speichern. Abhängig vom Wert dieser Variable sollte der Treiber-Bot eine andere Aktion durchführen, um den entsprechenden Pfad auf deiner Schatzkarte zu durchlaufen. Sobald der Fahrer-Bot an der nächsten Kreuzung angekommen ist, muss der Roboter dem Navigator-Bot signalisieren, dass er für den nächsten Befehl bereit ist.

Benutze deinen Plan, um das Treiberbot-Programm zu schreiben. Sobald der Treiberroboter richtig programmiert ist, sollte er in der Lage sein, den IR-Meldungen des Navigatorroboters zu folgen und zu jeder Position auf der Schatzkarte zu gelangen, egal welche es ist.

Name_____

Probiere es aus!

Sobald du deine Schatzkarte fertig hast und dein Driver-Bot programmiert ist, wähle einen Ort auf deiner Karte aus, an dem der Schatz liegen soll. Programmiere deinen Navigator-Bot so, dass er die IR-Meldungen in der richtigen Reihenfolge sendet, damit dein Fahrer-Bot den Schatz erreicht. Achte darauf, dass der Navigatorbot zwischen den einzelnen Befehlen auf das Signal des Fahrerroboters wartet:

Lasse deine Programme in deinen Robotern laufen, um sie zu testen. Bist du am richtigen Schatz angekommen? Wenn nicht, versuche herauszufinden, was schief gelaufen ist. Passe deine Programme an und versuche es noch einmal. Teste so lange, bis du deine Programme zum Laufen gebracht hast.

Wenn du am Schatz angekommen bist, wähle einen neuen Ort auf deiner Karte aus, um den neuen Standort des Schatzes zu bestimmen, und teste noch einmal.

Nicht vergessen

Sobald der Treiber-Bot richtig programmiert ist, sollte er in der Lage sein, den IR-Meldungen des Navigator-Bots zu folgen und zu jeder Schatzkartenposition zu gelangen, egal welche es ist. Du solltest nur die Reihenfolge der Befehle in deinem Navigatorbot-Programm ändern müssen, um den Driverbot von Anfang an an jeden beliebigen Schatzort zu schicken.

Name_____

U5-1.4d: Der Edison-Refrain

Edisons Infrarot (IR)-Lichtsensor ist der Sensor, der Edison Infrarotlicht ausstrahlen und erkennen lässt. Wir können den IR-Sensor benutzen, um Nachrichten an andere Edison-Roboter zu senden oder Nachrichten von ihnen zu empfangen. Ein Roboter kann mit seinen beiden IR-LEDs eine Infrarot-Nachricht aussenden, die der IR-Empfänger eines anderen Roboters erkennen kann.

Nicht vergessen

Infrarot wird manchmal als IR abgekürzt. In EdScratch sind IR-Meldungsblöcke Blöcke, die sich auf Edisons Infrarot-Meldungen mit Hilfe der Infrarot-LEDs und des Infrarot-Empfängers beziehen.

Um eine Infrarot-Nachricht mit deinem Edison-Roboter in EdScratch zu versenden, musst du den send IR messages-Block verwenden, der einen Eingabeparameter hat, der es dir erlaubt, eine bestimmte Nachricht zu senden. Du kannst die Nachricht ändern, indem du den Wert des Eingabeparameters im Sende-IR-Nachrichtenblock änderst. Der Sende-IR-Meldungsblock hat einen Eingabebereich von 0 bis 255. Mit anderen Worten, Edison kann 256 verschiedene 'Nachrichten' senden und empfangen.

Indem wir IR-Nachrichten mit unterschiedlichen Werten verwenden, können wir ein Programm erstellen, das Edison anweist, auf verschiedene Weise zu reagieren, je nachdem, welche IR-Nachricht er erkennt. Wir können auch verschiedene Edison-Roboter dazu bringen, nur auf bestimmte IR-Nachrichten zu reagieren.

Damit das funktioniert, müssen wir eine Variable verwenden, die die Daten speichert, die Edison mit seinem IR-Sensor empfängt.

Versuchen wir, die IR-Meldung zu benutzen, um mehrere Edison-Roboter zu koordinieren, damit sie ein Lied in einer Runde zusammen spielen können.

Nicht vergessen

Eine Variable ist ein Stück Speicher, das verwendet wird, um einige Informationen in einem Programm zu speichern. Da alle Sensoren von Edison Daten als Werte an Edison zurücksenden, kannst du diesen Wert in einer Variablen speichern.

Name_____

Was zu tun ist

Ein Kanon ist ein Musikstück, bei dem zwei oder mehr Personen (oder Roboter!) die gleiche Melodie singen oder spielen, aber jede beginnt zu einer anderen Zeit. Während jeder Teilnehmer einen anderen Teil des Liedes singt oder spielt, harmonisiert die Melodie sie dennoch miteinander. Bei dieser Aktivität wirst du mehrere Edison-Roboter einsetzen, um eine Melodie in einem Kanon zu spielen.

Für diese Aktivität müsst ihr in einer Gruppe mit mindestens drei Edison-Robotern arbeiten. Einer der Edison-Roboter muss der Dirigentenroboter sein, und alle anderen Roboter sind Darstellerroboter. Der Dirigentenroboter verwendet IR-Nachrichten, um die Darsteller-Bots zu verfolgen und jedem Darsteller-Bot zu sagen, wann er Musik spielen soll.

Dieses Projekt besteht aus drei Teilen: Auswahl und Arrangement deines Songs, Planung deiner IR-Nachrichten und Programmierung aller Roboter. Als erstes müsst ihr ein Lied auswählen, das die Roboter spielen sollen.

Wähle und arrangiere dein Lied

Zusammen mit eurer Gruppe müsst ihr wählen, welches Lied ihr aufführen wollt. Viele Kinderlieder funktionieren gut in einem Kanon, wie z.B. Row, Row, Row Your Boat. Ihr müsst auch entscheiden, an welchen Stellen im Lied ihr jeden neuen Darsteller Bot mitmachen lasst.

1. Welches Lied werden eure Darsteller-Bots spielen und wann wird jeder neue Darsteller-Bot mitmachen? Schreibe deinen Plan auf, um dir zu helfen, wenn du jeden Darsteller-Bot programmieren willst, dein Lied zu spielen.

_____

_____

_____

_____

Optional: Um deinen Song in EdScratch zu programmieren, musst du jede einzelne Note ausschreiben. Wenn du möchtest, kannst du diesen Platz nutzen, um deine Musik aufzuschreiben, damit du sie leichter in deine Performer-Bots einprogrammieren kannst.

Name_____

Der IR-Nachrichtenplan

Einer deiner Edison-Roboter wird dein Dirigentenroboter sein. Der Dirigenten-Bot wird das Lied nicht spielen, sondern stattdessen alle Darsteller-Bots mit Hilfe von IR-Meldungen verwalten. Der Dirigenten-Bot muss sowohl IR-Nachrichten senden als auch empfangen.

Der Dirigenten-Bot muss Nachrichten senden, um jedem Darsteller-Bot zu signalisieren, dass er anfangen soll, ein Lied nach dem anderen zu spielen. Wenn ein Darsteller-Bot an die Stelle im Lied gelangt, an der der nächste Roboter mitspielen soll, muss der Darsteller-Bot dem Dirigenten-Bot eine andere IR-Nachricht senden. Der Dirigenten-Bot wird dann wissen, dass der nächste Darsteller-Bot spielen soll.

Dadurch entsteht ein Muster in den IR-Meldungen, die der Dirigenten-Bot sendet und empfängt. Der Conductor-Bot muss dem Conductor-Bot eine IR-Meldung senden, damit der erste Darsteller-Bot spielen kann, auf das Signal von diesem Darsteller-Bot warten und dann die nächste IR-Meldung senden, damit der nächste Darsteller spielen kann. Der Conductor-Bot muss dann dieses Muster des Wartens auf eine IR-Meldung wiederholen, prüfen, von welchem Darsteller diese Meldung kam, und dann immer wieder den nächsten Darsteller signalisieren, bis alle Darsteller-Bots zu spielen begonnen haben.

Name_____

Wie viele verschiedene Nachrichten der Dirigenten-Bot senden und empfangen können muss, hängt davon ab, wie viele Darsteller-Bots in deinem Chor sind. Jede der von dir gesendeten IR-Nachrichten muss einen anderen Wert haben, damit die empfangenden Roboter sie unterscheiden können. Du kannst für deine IR-Nachrichten beliebige Werte zwischen 0 und 255 verwenden. Achte darauf, dass deine Nachrichten in allen Programmen deiner Roboter übereinstimmen.

2. Wie viele Nachrichten muss euer Dirigenten-Bot senden und empfangen können? Welchen Wert wirst du für jede Nachricht verwenden? Für welchen Roboter ist jede Nachricht bestimmt und was muss der empfangende Roboter tun, wenn er die IR-Nachricht entdeckt hat? Arbeitet zusammen und entwerft euren Plan. Macht euch Notizen, die euch beim Schreiben eurer EdScratch-Programme helfen sollen.

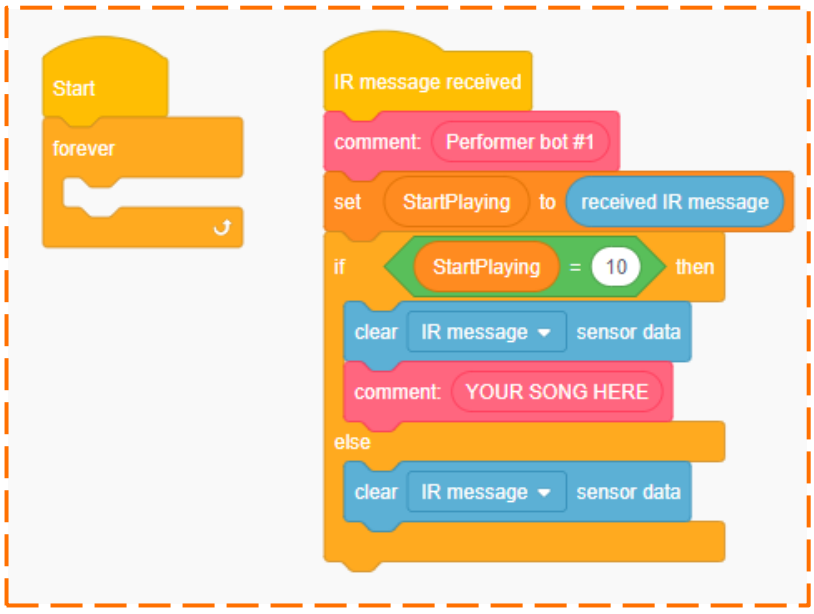
Wert der IR-Meldung	Edison schicken (Dirigent oder welcher Darsteller)	Edison empfangen (Dirigent oder welcher Darsteller)	Edisons Aktion empfangen

Programmiere alle Roboter

Alle Edison-Roboter, die das Lied spielen werden, sind die Darsteller-Bots. Das Programm jedes Darsteller-Bots sagt dem Roboter, dass er mit dem Spielen warten soll, bis er eine IR-Nachricht vom Dirigenten-Bot erhält. Da wir nicht wollen, dass alle Roboter gleichzeitig anfangen zu spielen, muss jeder Roboter nur dann mit dem Spielen beginnen, wenn er die ihm zugewiesene IR-Meldung erkennt.

Name_____

Seht euch den folgenden Code an:



Dies ist kein vollständiges Programm, aber du kannst diesen Code als Beispiel verwenden, um die Programme deiner Darsteller-Bots zu erstellen. Dieser Abschnitt des Codes verwendet eine Variable namens StartPlaying. Der Wert dieser Variable ist so eingestellt, dass er dem Wert der empfangenen IR-Meldung entspricht. Wenn du mit Sensorwerten arbeitest, solltest du alle Werte, die du in deinem Programm verwenden möchtest, immer in einer Variablen speichern. Dann kannst du dein Programm diese Variable überall im Programm auf jede beliebige Art und Weise verwenden lassen.

Woran liegt das?

Computer verfolgen die Werte von Sensoren nicht, es sei denn, du sagst es ihnen. Wenn du einem Computer sagst, er soll einen Wert in einer Variablen speichern, sagst du ihm, er soll sich diesen Wert merken.

Erinnere dich daran, dass der IR-Detektor von Edison immer eingeschaltet ist und immer wieder nach Daten sucht. Der Roboter merkt sich nicht ewig jeden Wert, den er erkennt. Er behält den Wert nur so lange in seinem Arbeitsspeicher, bis er in einem Programm aufgerufen oder durch einen neuen Messwert ersetzt wird. Indem du den Wert der IR-Meldung in einer Variablen speicherst, sagst du Edison, dass er diesen Wert festhalten soll, damit du etwas damit machen kannst.

Der Wert bleibt so lange in der Variable, bis der Sensor eine neue IR-Meldung erkennt und der eingestellte Block den alten Wert durch den neuen ersetzt.

Name_____

Jedes der Programme deiner Darsteller-Bots muss nach einer IR-Meldung suchen und den Wert dieser Meldung dann als Variable speichern. Je nach dem Wert dieser Variable sollte der Darsteller-Bot entweder anfangen zu spielen oder einfach die Daten löschen und wieder auf seine IR-Meldung warten.

Benutze deinen Plan, um ein Programm für jeden deiner Performer-Bots zu schreiben.

Tipp!

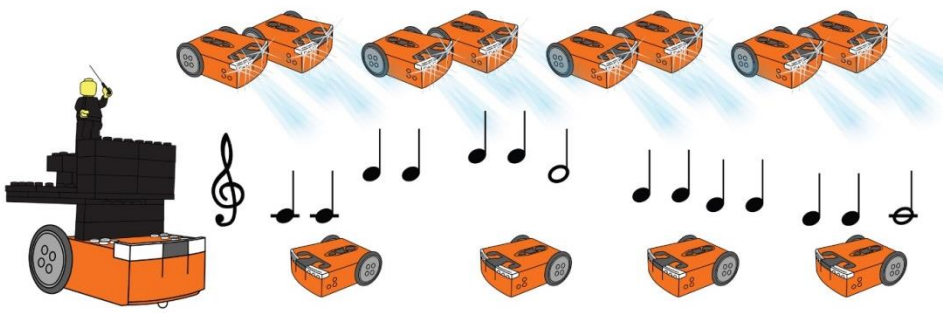
Achte darauf, dass jeder Darsteller-Bot den richtigen send-IR-message-Block an der richtigen Stelle im Song hat!

Du musst auch den Conductor-Bot programmieren, indem du deinen Plan als Anleitung verwendest. Denke daran, dass dein Conductor-Bot auch nach einer IR-Meldung suchen muss und dann den Wert dieser Meldung als Variable speichern muss. Je nach Wert der Variable muss der Conductor-Bot eine andere IR-Meldung senden, damit der nächste Darsteller-Bot starten kann.

Tipp!

Immer wenn du IR-Daten empfängst und diesen Wert als Variable speicherst, solltest du die IR-Daten löschen, indem du als nächste Aktion in deinem Programm den Datenblock IR-Meldung löschen verwendest.

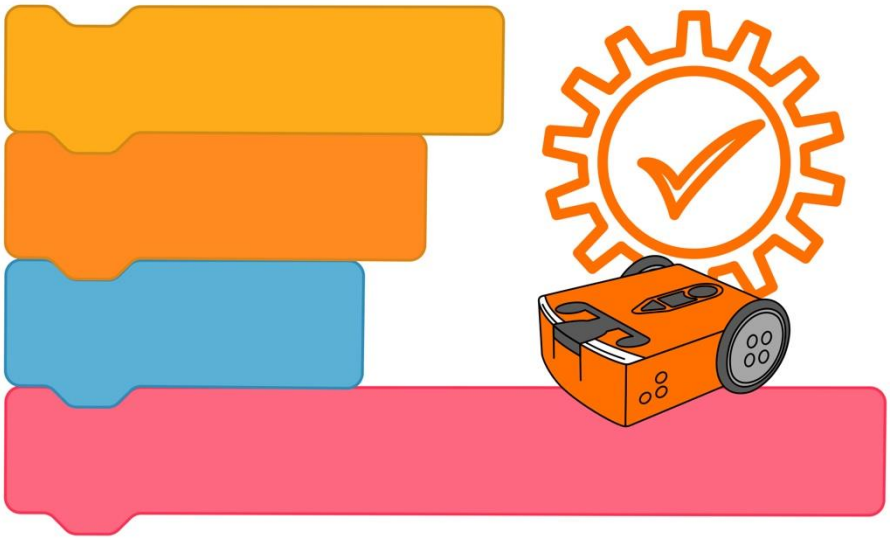
Sobald alle eure Bots programmiert sind, testet eure Programme mit euren Robotern. Hat jeder Roboter gespielt, wann er sollte? Wenn nicht, versuche herauszufinden, was schief gelaufen ist. Passe deine Programme an und versuche es erneut. Teste weiter, bis du deine Programme zum Laufen gebracht hast.



Name_____

Arbeitsblatt U5-1: Vier Zeilen

Einheit 6: Zeit des Erfinders!



Edison

Name_____

U6-1.1: Design-Build-Test-Zyklus erkunden

Eines der besten Dinge am Lernen von Informatik ist, dass es nicht nur darum geht, Programme zu schreiben. Du kannst Informatik benutzen, um alle möglichen Dinge zu erstellen, einschließlich verschiedener Erfindungen mit deinen Edison-Robotern.

Inzwischen weißt du, wie Edison-Roboter funktionieren und verschiedene Dinge, die du mit Edison machen kannst, indem du den Roboter in EdScratch programmierst. Das bedeutet, dass du die Fähigkeiten hast, ein Erfinder zu sein, der Edison und EdScratch benutzt, um alle möglichen Kreationen zu machen und zu programmieren.

Was könntest du erschaffen? Wie könntest du ihn bauen? Wie würde das Computerprogramm, das du für deine Kreation brauchst, aussehen?

Etwas mit Edison und EdScratch zu erfinden, ist eine große Aufgabe, aber du kannst es viel leichter handhaben, indem du es in kleinere Stücke zerlegst und jedes Ding einzeln machst.

Fachjargon

In der Informatik ist die Zersetzung (decomposition) der Prozess, ein komplexes Problem oder System in Teile zu zerlegen, die leichter zu verstehen, zu verwalten und zu lösen sind. Ein großes Problem in kleinere, leichter lösbare Aufgaben zu zerlegen, macht es einfacher, genau zu bestimmen, was passieren muss und in welcher Reihenfolge die einzelnen Aufgaben erledigt werden müssen.

Die Zerlegung zu benutzen, um jede größere Aufgabe in kleinere Teile zu zerlegen, ist einer der besten Wege, sich ihr zu nähern. Wenn du feststeckst oder dich überfordert fühlst, wenn du ein Projekt in Angriff nimmst, nimm dir eine Minute Zeit, um darüber nachzudenken, wie du die Zerlegung nutzen kannst, um es in kleinere Aufgaben zu zerlegen.

Wenn es darum geht, deine eigenen Erfindungen mit Edison zu kreieren, kannst du auch iteratives Testen durch etwas verwenden, das Design-Build-Test-Zyklus genannt wird, um dir zu helfen.

Name_____

Fachjargon

Der Design-Build-Test-Test-Zyklus, der auch manchmal Design-Build-Test-Lernzyklus genannt wird, ist ein Prozess, bei dem man etwas entwirft, baut und dann ausprobiert, um zu sehen, was funktioniert und was verbessert werden muss. Dann nimmst du das, was du aus dem Test gelernt hast und wendest es auf das Design an. Weil es ein Zyklus ist, wiederholst du jeden Schritt und wendest das Gelernte an, um den nächsten Zyklus besser zu machen, bis du mit dem Ergebnis zufrieden bist.

Viele Versionen von etwas zu machen und zu testen, das Gelernte anzuwenden und jedes Mal Änderungen vorzunehmen, wird als iteratives Testen bezeichnet und ist eine gängige Praxis bei der Entwicklung neuer Dinge in der Informatik. Technologieunternehmen verwenden den Design-Build-Test-Zyklus und das iterative Testen immer dann, wenn sie neue Produkte entwickeln.

Es ist ziemlich unwahrscheinlich, dass dein erster Entwurf perfekt funktionieren wird. Das ist schon in Ordnung! Wenn du iteratives Testen verwendest und den Design-Build-Test-Zyklus durcharbeitest, kannst du deine Idee immer weiter verfeinern, das Gelernte anwenden und Verbesserungen vornehmen.

Denke daran, dass ein großer Teil der Informatik rechnergestütztes Denken ist, und ein großer Teil des rechnergestützten Denkens ist das Lösen von Problemen!

Nicht vergessen

Computernahes Denken bedeutet, über ein Problem oder eine Aufgabe in ähnlicher Weise nachzudenken, wie ein Computer denkt. Es ist ein Weg, Probleme logisch durchzuarbeiten, sie in kleinere Stücke zu zerlegen, Muster zu suchen und dann die Informationen zu nutzen, um eine Schritt-für-Schritt-Lösung zu finden.

Mit anderen Worten, computergestütztes Denken ist eine Art und Weise, Informationen zu planen, Probleme zu lösen und zu analysieren, ähnlich wie es ein Computer tut.

Name_____

Ein anderer großer Teil der Informatik besteht darin, kreativ zu sein und neue Dinge auszuprobieren. Du weißt nie, was du vielleicht erschaffen kannst, bis du es ausprobierst!

Aufgabe 1: Brainstorming

Bevor du anfangen kannst, etwas zu entwerfen, zu bauen und zu testen, musst du dir ein paar Ideen einfallen lassen. Bei der Ideenfindung geht es darum, fantasievoll zu sein, deine Kreativität zu nutzen und über Möglichkeiten nachzudenken.

Brainstorming ist eine Möglichkeit, dein Denken anzukurbeln, um Ideen zu generieren. Wenn du ein Brainstorming machst, versuchst du, die Gedanken frei fließen zu lassen. Alle Ideen sind akzeptabel, egal, was sie sind. Triff beim Brainstorming keine Entscheidungen darüber, ob die Ideen möglich sind, oder beurteile die Ideen.

Lass uns versuchen, ein paar Edison-Erfindungen zu brainstormen. Mit Hilfe des Arbeitsblattes U6-1 musst du dir sechs verschiedene Ideen für etwas einfallen lassen, das du mit Edison-Robotern erstellen und programmieren kannst. Für jede Idee hast du nur 45 Sekunden Zeit, bevor du zu deiner nächsten Idee übergehen musst.

Nimm deine Ideen so auf, wie du willst. Du kannst etwas zeichnen, die Idee in Worte fassen oder ein bisschen von beidem machen! Es ist dir allerdings nicht erlaubt, NICHT mit Ideen zu kommen. Denkt daran, dass es während einer Brainstorming-Sitzung keine 'schlechten' Ideen gibt!

Stelle einen Timer für 45 Sekunden ein und fange mit deiner ersten Idee an. Sobald der Timer abgelaufen ist, stellst du ihn zurück und machst mit der nächsten Idee weiter, bis du alle sechs Ideen fertig hast.

Nachdem du das Brainstorming beendet hast, analysiere die Ideen, die du dir ausgedacht hast. Wenn ihr euch alle eure Ideen zusammen anseht, entscheidet ihr vielleicht, dass einige eurer Ideen besser sind als andere. Du kannst dann eine dieser Ideen auswählen und mit der Gestaltung deiner Kreation fortfahren.

Aufgabe 2: Entwerfen

Sobald du mindestens eine Idee hast, von der du denkst, dass sie funktionieren könnte, kannst du mit der Planung und Gestaltung beginnen. Ihr müsst eure Idee in kleinere Teile zerlegen, jedes Teil entwerfen und planen, wie ihr jedes einzelne in Angriff nehmen wollt. Zumindest solltest du deine Idee in zwei Teile zerlegen: den physischen Entwurf mit Edison und den Programmentwurf.

Es gibt viele verschiedene Werkzeuge, die dir beim Entwerfen und Planen helfen können. Du kannst Skizzen oder Diagramme zeichnen, du kannst ein Storyboard erstellen oder eine Übersicht schreiben. Wenn du dein Programm entwirfst, wirst du wahrscheinlich auch Pseudocode verwenden wollen, um zu planen, wie das Programm funktionieren soll. Du kannst auch immer einen Mix aus verschiedenen Ansätzen verwenden - was immer für dich am besten funktioniert!

Benutze eine deiner Ideen und arbeite einen Entwurf für die physische Schöpfung und das Programm aus.

Hier sind einige der Dinge, über die du beim Entwerfen nachdenken solltest:

Name_____

- Was wird deine Schöpfung tun?
- Wie wird deine Schöpfung aussehen?
- Welche Materialien könntest du für den Bau deiner Kreation verwenden? Wie wirst du diese an Edison anbringen?
- Wie wird das Programm Edison kontrollieren, damit deine Schöpfung funktioniert?

Denke daran, dass dieses Design nur dein erstes ist. Es gibt noch eine Menge zu lernen! Es kann sein, dass dein Design nicht alles vorhersagen kann, und du hast vielleicht ein paar Dinge, bei denen du dir nicht sicher bist. Wenn du Fragen dazu hast, wie Teile deiner Kreation oder deines Programms funktionieren können, schreib dir diese Fragen auf. Das sind die Dinge, auf die du beim Bauen und Testen besonders achten solltest.

1. Gib deinem Projekt einen Namen und schreibe eine kurze Beschreibung deiner Idee.

_____

_____

_____

_____

2. Was sind die verschiedenen Teile, in die du dein Projekt zerlegt hast, um dir beim Design zu helfen? Denke daran, dass du mindestens zwei Teile haben solltest: die physische Erstellung mit Edison und das Programmdesign.

_____

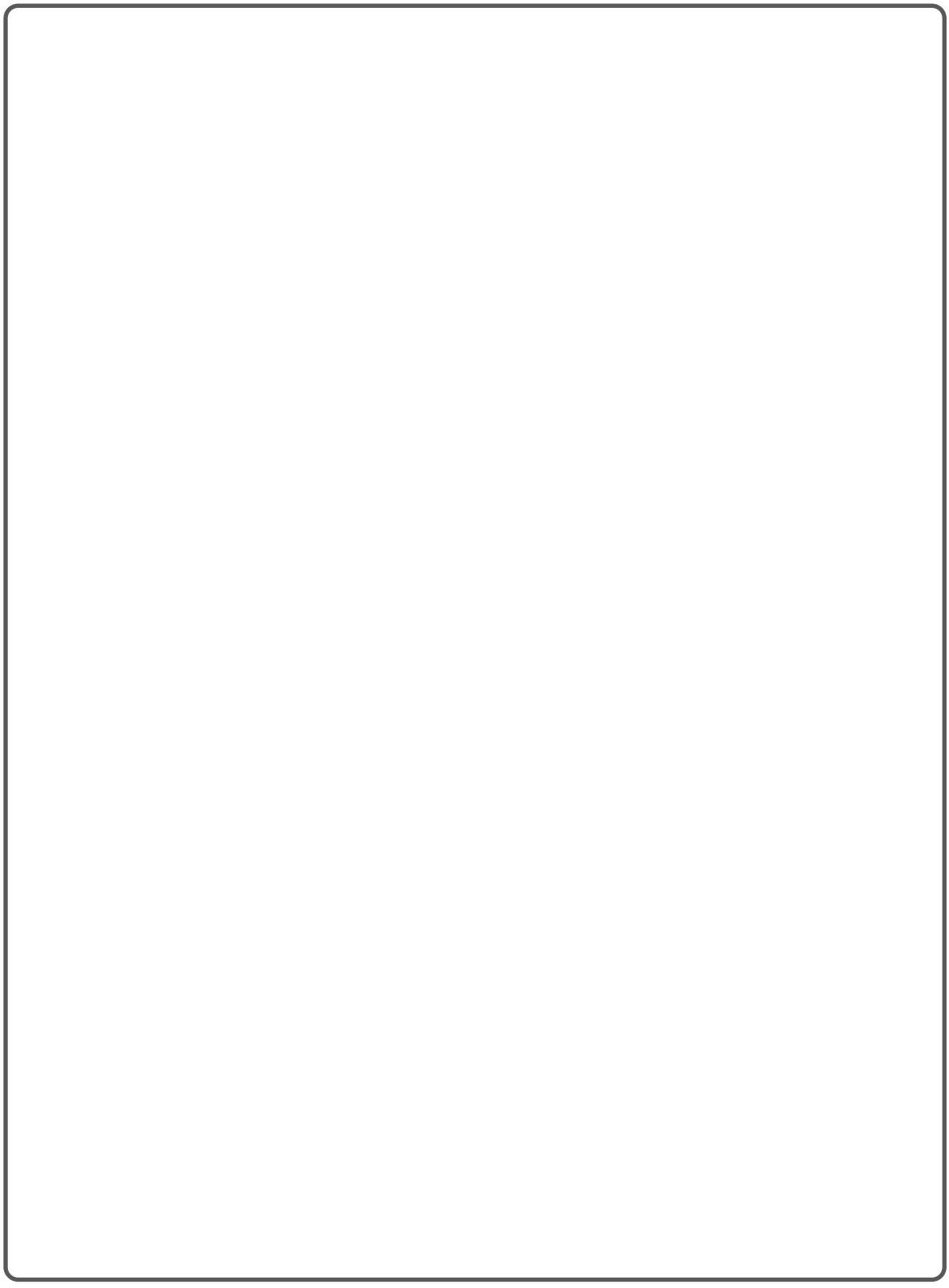
_____

_____

_____

Name_____

3. Arbeite das Design deiner physischen Schöpfung aus. Benutze extra Platz, wenn du ihn brauchst.



4. Entwerfe dein EdScratch Programm für deine Kreation. Benutze extra Platz, wenn du ihn brauchst.

Name_____

Optional: Was konntest du bei deinem Design nicht ausarbeiten? Auf welche Fragen musst du besonders achten, wenn du baust und testest? Notiere sie in diesem Feld.

Name_____

Mini-Herausforderung!

Ist deine Idee möglich? Kann sie gebaut werden? Kann sie programmiert werden? Der einzige Weg, das herauszufinden, ist, es zu versuchen!

Versuche, deine Idee in die Realität umzusetzen, indem du deinen Entwurf als Leitfaden verwendest, der dir beim Bauen und Programmieren deiner Kreation hilft. Teste deine Erfindung, um zu sehen, was funktioniert und was nicht. Nimm das, was du aus deinem Test gelernt hast, und wende es wieder auf deinen Entwurf an. Wiederhole den Zyklus immer wieder und wende das Gelernte an, um bei jeder Iteration Verbesserungen vorzunehmen.

Name_____

U6-1.1a: Erfinde ein Fantasiewesen

Lebt es unter dem Bett und frisst dreckige Socken? Ist es gemein und furchterregend oder schüchtern, aber nett? Hat es 23 Beine und eine schöne Singstimme? Schießt es herum oder bewegt es sich wie eine Schnecke?

Bei dieser Herausforderung hängt alles von dir ab!

Was du tun musst

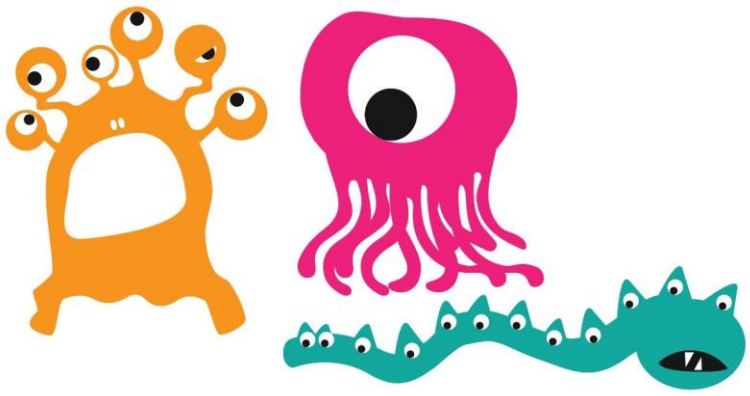
Für diese Herausforderung musst du den Design-Build-Test-Zyklus verwenden, um eine imaginäre Kreatur mit deinem Edison-Roboter und EdScratch zu erstellen und zu programmieren. Wie deine Kreatur aussieht und wie sie funktioniert, ist dir überlassen, aber deine Kreatur muss die folgenden Dinge tun können:

- ☐ bewege dich mit Edisons Motorausgängen
- ☐ Benutze mindestens einen von Edisons Sensoren, um ein Verhalten deiner Kreatur auszulösen, das auf etwas in ihrer Umgebung reagiert.

Verbringe etwas Zeit mit dem Brainstorming verschiedener Ideen. Sobald du mindestens eine Idee hast, von der du denkst, dass sie funktionieren könnte, fahre mit dem Entwerfen fort. Zerlege deine Idee in kleinere Teile, entwerfe jedes Teil und plane, wie du jedes einzelne in Angriff nehmen kannst. Zumindest solltest du deine Idee in zwei Teile zerlegen: das physische Design mit Edison und das EdScratch-Programmdesign.

Arbeite ein Design sowohl für den physischen Entwurf als auch für das Programm aus. Baue deine Kreatur und schreibe das EdScratch-Programm für deine Kreatur. Teste deine Erfindung, um zu sehen, was funktioniert und was verbessert werden muss.

Wende das, was du aus deinem Test gelernt hast, auf deinen Entwurf an, erstelle die nächste Iteration deines Builds und teste erneut. Wiederhole den Design-Build-Test-Zyklus so lange, bis deine Kreatur so funktioniert, wie du willst.



Name_____

U6-1.1b: Erfinde einen Wattebausch-Werfer

Schnapp dir einen Sack Wattebällchen, dann mach dich bereit, zielen und FEUERN! Es ist Zeit, einen Wattebausch-Werfer zu bauen. Werden deine Wattebällchen wie Raketen nach oben schießen? Wirst du ins Schwarze treffen können? Oder werden deine Wattebällchen den Flur hinunterfliegen?

Bei dieser Herausforderung hängt alles von dir ab!

Was du tun musst

Für diese Herausforderung musst du den Design-Build-Test-Zyklus verwenden, um einen Wattebausch-Werfer mit deinem Edison-Roboter und EdScratch zu erstellen und zu programmieren. Wie dein Raketenwerfer aussieht und wie er funktioniert, ist dir überlassen, aber dein Raketenwerfer muss mindestens eines der folgenden Dinge tun können:

- ☐ einen Wattebausch so hoch wie möglich werfen, oder
- ☐ einen Wattebausch so weit wie möglich werfen, oder
- ☐ werfe einen Wattebausch so genau wie möglich

Verbringe etwas Zeit mit dem Brainstorming verschiedener Ideen. Sobald du mindestens eine Idee hast, von der du denkst, dass sie funktionieren könnte, machst du mit dem Design weiter. Zerlege deine Idee in kleinere Teile, entwerfe jedes Teil und plane, wie du jedes einzelne in Angriff nehmen willst. Zumindest solltest du deine Idee in zwei Teile zerlegen: das physische Design mit Edison und das EdScratch-Programmdesign.

Arbeite ein Design sowohl für den physischen Entwurf als auch für das Programm aus. Baue deinen Raketenwerfer und schreibe dein EdScratch-Programm. Teste deine Erfindung, um zu sehen, was funktioniert und was verbessert werden muss.

Wende das, was du aus deinem Test gelernt hast, auf deinen Entwurf an, erstelle die nächste Iteration deines Builds und teste erneut. Wiederhole den Design-Build-Test-Zyklus so lange, bis dein Raketenwerfer so funktioniert, wie du willst.

Name_____

U6-1.1c: Erfinde einen Einbrecheralarm

Dir wurde ein wertvoller Schatz anvertraut, den es zu schützen gilt. Aber es gibt hinterhältige Diebe, die ihn sich holen wollen! Wie willst du den Schatz sicher verwahren?

Bei dieser Herausforderung liegt es an dir!

Was ihr tun müsst

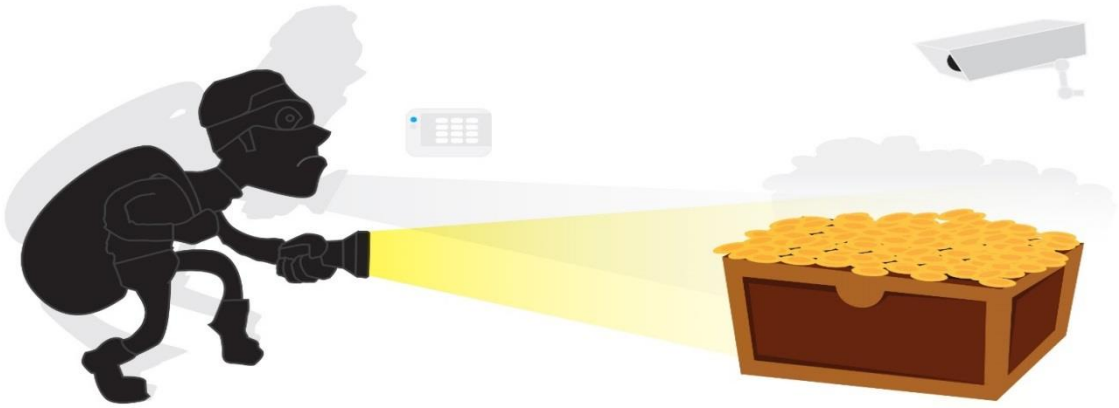
Für diese Herausforderung musst du den Design-Build-Test-Zyklus verwenden, um mit deinem Edison-Roboter und EdScratch einen Einbruchalarm zu erstellen und zu programmieren. Wie deine Erfindung aussieht und wie sie funktioniert, ist dir überlassen, aber deine Alarmanlage muss die folgenden Dinge tun können:

- ☐ benutze mindestens einen von Edisons Sensoren, um den Alarm auszulösen
- ☐ Tu etwas, um den Eindringling zu verscheuchen

Verbringe etwas Zeit mit dem Brainstorming verschiedener Ideen. Sobald du mindestens eine Idee hast, von der du denkst, dass sie funktionieren könnte, machst du mit dem Entwerfen weiter. Zerlege deine Idee in kleinere Teile, entwerfe jedes Teil und plane, wie du jedes einzelne in Angriff nehmen willst. Zumindest solltest du deine Idee in zwei Teile zerlegen: das physische Design mit Edison und das EdScratch-Programmdesign.

Arbeite ein Design sowohl für den physischen Entwurf als auch für das Programm aus. Baue deine Alarmanlage und schreibe dein EdScratch-Programm. Teste deine Erfindung, um zu sehen, was funktioniert und was verbessert werden muss.

Wende das, was du aus deinem Test gelernt hast, auf deinen Entwurf an, erstelle die nächste Iteration deines Builds und teste erneut. Wiederhole den Design-Build-Test-Zyklus so lange, bis dein Einbrecheralarm so funktioniert, wie du willst.



Name_____

U6-1.1d: Erfinde eine Mausefalle

Leg den Käse aus! Oder bevorzugen Mäuse Erdnussbutter? Was auch immer du als Köder benutzen willst, es ist Zeit, sich darauf vorzubereiten, eine Maus zu fangen! Was wird die Falle auslösen? Wie wirst du wissen, ob du eine Maus gefangen hast?

Bei dieser Herausforderung liegt es an dir!

Was ihr tun müsst

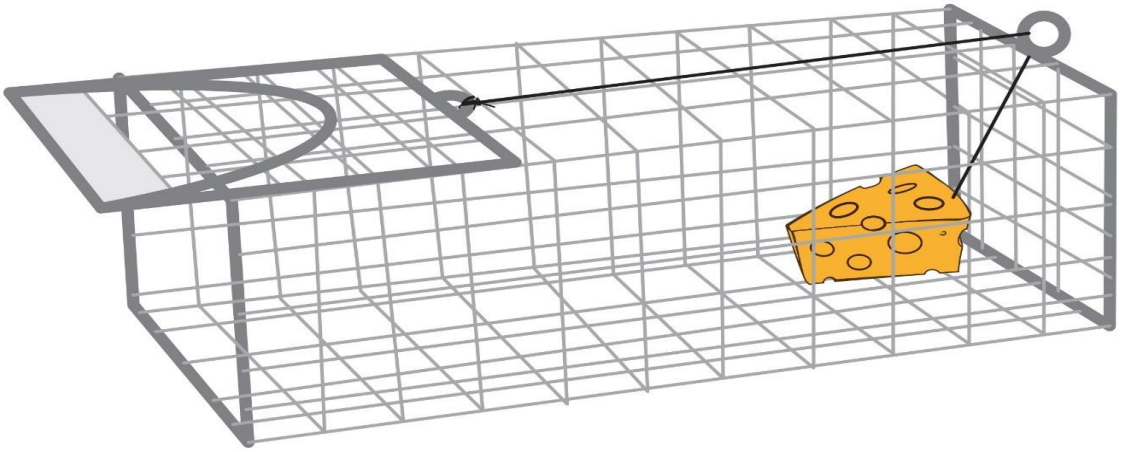
Für diese Herausforderung musst du den Design-Build-Test-Zyklus verwenden, um eine Mausefalle mit deinem Edison-Roboter und EdScratch zu erstellen und zu programmieren. Wie deine Erfindung aussieht und wie sie funktioniert, ist dir überlassen, aber deine Mausefalle muss die folgenden Dinge tun können:

- ☐ benutze mindestens einen von Edisons Sensoren, um die Falle auszulösen
- ☐ Tu etwas, um dich zu warnen, dass die Falle zugeschnappt wurde.

Verbringe etwas Zeit mit dem Brainstorming verschiedener Ideen. Sobald du mindestens eine Idee hast, von der du denkst, dass sie funktionieren könnte, machst du mit dem Entwerfen weiter. Zerlegt eure Idee in kleinere Teile, entwerft jedes Teil und plant, wie ihr jedes Teil angehen wollt. Zumindest solltest du deine Idee in zwei Teile zerlegen: das physische Design mit Edison und das EdScratch-Programmdesign.

Arbeite ein Design sowohl für den physischen Entwurf als auch für das Programm aus. Baue deine Mausefalle und schreibe dein EdScratch-Programm. Teste deine Erfindung, um zu sehen, was funktioniert und was verbessert werden muss.

Wende das, was du aus deinem Test gelernt hast, auf deinen Entwurf an, erstelle die nächste Iteration deines Builds und teste erneut. Wiederhole den Design-Build-Test-Zyklus so lange, bis du deine Falle genau wie gewünscht zum Funktionieren bringst.



Name_____

U6-1.1e: Erfinde einen Kombinationssafe

Wo bewahrst du deine wertvollsten Besitztümer auf? Warum packst du deine Habseligkeiten nicht in einen Kombinationssafe mit einem Roboterschloss! Wie wird dein Safe aussehen? Wie wirst du den Code eingeben, um in den Safe zu gelangen? Wie wird der Code lauten?

Bei dieser Herausforderung hängt alles von dir ab!

Was du tun musst

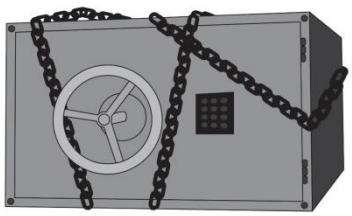
Für diese Herausforderung musst du den Design-Build-Test-Zyklus verwenden, um einen Kombinationssafe zu erstellen und zu programmieren, der deinen Edison-Roboter als Schloss verwendet. Wie deine Erfindung aussieht und wie sie funktioniert, ist dir überlassen, aber dein Schloss darf deinen Safe nur öffnen, wenn die richtige Kombination eingegeben wird. Dein Schloss sollte eines der folgenden Dinge tun:

- ☐ nur für die richtige Abfolge von runden und dreieckigen Knopfdrücken öffnen, oder
- ☐ nur für die richtige Sequenz von TV- oder DVD-Fernbedienungstastendrücken öffnen, oder
- ☐ nur für die richtige Sequenz von IR-Nachrichten von einem anderen Edison-Roboter geöffnet, oder
- ☐ nur für die richtige Kombination aus einer Mischung von Tastendrücken, Fernbedienungssignalen oder IR-Meldungen geöffnet

Verbringe etwas Zeit mit dem Brainstorming verschiedener Ideen. Sobald du mindestens eine Idee hast, von der du denkst, dass sie funktionieren könnte, machst du mit dem Entwerfen weiter. Zerlege deine Idee in kleinere Teile, entwerfe jedes Teil und plane, wie du jedes einzelne in Angriff nehmen kannst. Zumindest solltest du deine Idee in zwei Teile zerlegen: den physischen Entwurf mit Edison und den Entwurf für das EdScratch-Programm.

Arbeite ein Design sowohl für den physischen Entwurf als auch für das Programm aus. Baue deinen Kombinationssafe und schreibe dein EdScratch-Programm. Teste deine Erfindung, um zu sehen, was funktioniert und was verbessert werden muss.

Wende das, was du aus deinem Test gelernt hast, auf deinen Entwurf an, erstelle die nächste Iteration deines Builds und teste erneut. Wiederholt den Design-Build-Test-Zyklus so lange, bis euer Kombinationssafe so funktioniert, wie ihr wollt.



Name_____

U6-1.2: Lass uns ein Spukhaus erforschen

All die verschiedenen Sensoren und Ausgänge von Edison bedeuten, dass es eine Menge Dinge gibt, die man mit Edison-Robotern machen kann. Mit EdScratch kannst du Edison so programmieren, dass er fährt, musikalische Geräusche abspielt, Hindernisse erkennt, schwarzen Linien folgt und vieles, vieles mehr.

Du kannst Edison auch in andere Dinge verwandeln, indem du mit dem Roboter Erfindungen baust und Edisons Sensoren und Ausgänge auf verschiedene Art und Weise nutzt. Du hast zum Beispiel gesehen, wie du die Blöcke in der Kategorie Drive in EdScratch benutzen kannst, um Edison nicht herumzufahren, sondern stattdessen Kreationen mit Edisons Motoren anzutreiben. Genauso kannst du Edisons verschiedene Sensoren benutzen, um Dinge auf unerwartete Weise zu tun.

In diesem Projekt werdet ihr zusammenarbeiten und alles, was ihr über Edison und EdScratch gelernt habt, nutzen, um eure Roboter in Monsterjäger zu verwandeln. Ihr werdet eure Monsterjagd-Roboter einsetzen, um euch dabei zu helfen, verschiedene Geister in einem Spukhaus aufzuspüren und zu fangen.

Über das Spukhaus

Es gibt verschiedene Arten von Monstern, die in dem Spukhaus leben. Glücklicherweise hängt jede Art von Monster nur in einem Raum herum. Deine Aufgabe ist es, mit Edison Erfindungen zu entwerfen, zu erschaffen und zu programmieren, um all die verschiedenen Monster aufzuspüren und zu fangen, einen Raum nach dem anderen.

In einem Raum des Spukhauses gibt es Geister. Diese weißen Geister hängen auf einem schwarzen Boden herum. Du musst einen Edison-Roboter programmieren, der mit den Geistern durch den Raum fährt und jeden einzelnen lokalisiert, damit du hineinkommen und jeden einzelnen Geist entfernen kannst.

In dem Spukhaus gibt es noch zwei weitere Räume. Welche Monster sind in jedem dieser Räume? Wie wird Edison dir helfen, diese anderen Kreaturen aufzuspüren und sogar in die Falle zu locken? Dieser Teil hängt von dir ab!

Aufgabe 1: Entdecke die Geister

Dein Spukhaus hat einen Raum mit Geistern darin. Deine Aufgabe ist es, diesen Raum von allen Geistern zu befreien, indem du einen Edison-Roboter benutzt, der dir dabei hilft.

Für diese erste Aufgabe gibt es zwei Hauptteile. Erstens, du musst einen Testraum erstellen, um den Raum und die Geister darzustellen. Zweitens musst du Edison so programmieren, dass er alle Geister aufspürt und dich jedes Mal alarmiert, wenn ein Geist gefunden wird, damit du hineinkommen und ihn entfernen kannst.

Der Testraum

Ihr müsst einen Testraum erstellen, um den Raum und die Geister darzustellen. Dein Raum braucht Wände und einen schwarzen Boden. Du brauchst auch etwas, um die Geister zu repräsentieren, wie zum Beispiel weißes Klebeband. Du musst mindestens drei Geister in deinem Zimmer haben.

Name_____

Das Programm

Normalerweise benutzen wir Edisons Linienverfolgungssensor, um schwarze (nicht reflektierende) auf einem weißen (reflektierenden) Hintergrund zu erkennen. Dieses Mal musst du genau das Gegenteil tun: finde weiße 'Geister' auf einer schwarzen Oberfläche.

Du musst ein Programm schreiben, damit Edison um den Testraum herumfährt. Wenn Edison ein Gespenst entdeckt, sollte der Roboter aufhören, sich zu bewegen und das Gespenst unter ihm 'einfangen'. Edison muss dann etwas tun, z.B. eine Melodie spielen, um euch wissen zu lassen, dass der Roboter einen Geist entdeckt hat. Dies wird euer Stichwort sein, hereinzukommen und den Geist aus dem Raum zu entfernen. Edison sollte warten, bis du den Geist, den er entdeckt hat, aus dem Raum geholt hast, und dann weiterfahren. Sobald Edison alle Geister entdeckt hat, sollte der Roboter dir signalisieren, dass alle Geister weg sind, und das Programm sollte beendet werden.

Tipp!

Edisons Linienverfolgungssensor funktioniert auf eine besondere Art und Weise. Erinnere dich daran, dass der Line Tracker die Menge an reflektiertem Licht misst, die er von der Unterseite des Roboters erfasst, und diese Messung als Wert speichert. Je mehr Licht erkannt wird, desto höher der Wert. Messwerte mit hohen Werten werden als 'reflektierend' für den Roboter angesehen.

Wenn der Line Tracker zum ersten Mal eingeschaltet wird, nimmt der Sensor eine kommt. Der Roboter verwendet diesen Anfangswert, um festzustellen, was 'reflektierend' Messung des reflektierten Lichts vor, das von der Oberfläche unter dem Roboter ist. Mit anderen Worten, der Roboter setzt den ersten Wert, den der Line Tracker erzeugt, auf den Wert 'reflektierend' und verwendet diesen, um zu bestimmen, ob ein neuer Wert 'reflektierend' oder 'nicht reflektierend' ist. Aus diesem Grund startet man ein Programm immer mit dem Zeilendetektor auf einer weißen Fläche.

In eurem Geisterjäger-Programm müsst ihr euer Programm auch auf einer weißen Oberfläche starten. Lege dein Edison auf ein Stück weißes Papier, bevor du dein Programm startest, damit der Roboter beim ersten Start deines Programms den Wert 'reflektierend' einstellen kann. Dein Programm sollte dann Edison warten lassen, bis du etwas tust, z.B. einen von Edisons Knöpfen drückst, um zum Rest des Programms überzugehen und nach Geistern zu suchen. Diese Zeit kannst du nutzen, um Edison vom weißen Papier auf die schwarze Oberfläche deines Testgebiets zu bewegen.

Entwerfe dein Geisterjagdprogramm, schreibe es in EdScratch und teste es dann mit deinem Edison-Roboter in deinem 'Geisterraum'-Testraum. Beachte, was in deinem Programm funktioniert und was verbessert werden muss. Wende das Gelernte auf dein Programmdesign an, schreibe deine Änderungen in EdScratch und teste es erneut.

Mache so lange weiter, bis du Edison dazu bringst, das Programm laufen zu lassen und alle Geister erfolgreich aufzuspüren.

Aufgabe 2: Erstelle deine anderen Räume im Spukhaus

Dein Spukhaus braucht zwei weitere Räume mit jeweils einer anderen Art von Monster in jedem Raum. Für jeden anderen Raum musst du entscheiden, welche Art von Monster sich in diesem Raum befindet und dann einen Testraum erstellen, der den Raum und die Monster repräsentiert. Du musst auch den Design-Build-Test-Zyklus verwenden, um einen Weg zu erstellen und zu programmieren, um alle Monster in jedem Raum mit mindestens einem Edison-Roboter und EdScratch aufzuspüren und einzufangen.

Verbringe einige Zeit damit, verschiedene Ideen darüber zu entwickeln, welche Monster du benutzen willst und wie du sie aufspüren und fangen kannst. Sobald du mindestens eine Idee hast, von der du denkst, dass sie funktionieren könnte, fahre mit dem Entwerfen fort. Zerlegt eure Idee in kleinere Abschnitte, entwerft jeden Teil und plant, wie ihr jeden einzelnen anpacken wollt. Mindestens solltest du deine Idee in drei Teile für jeden Raum zerlegen: den Testraum, die physische Erstellung mit Edison-Robotern und das EdScratch-Programmdesign.

Raum 1

Beantworte die Fragen und nutze diesen Raum, um den Raum, die Monster und deine Falle zu gestalten. Sobald du deinen Entwurf für jeden Teil hast, baue deinen Raum, die Monster und deine Monsterjagd-Erfindung. Programmiere deine Erfindung in EdScratch und teste dann deine Kreation in deinem Raum, um zu sehen, was funktioniert und was verbessert werden muss.

Wende das, was du aus deinem Test gelernt hast, auf dein Design an, erstelle die nächste Iteration deines Builds und teste es erneut. Wiederholt den Design-Build-Test-Zyklus so lange, bis ihr in der Lage seid, alle Monster aufzuspüren und einzufangen.

1. Welche Monster sind in diesem Raum? Wie viele Monster gibt es hier? Schreibe eine kurze Beschreibung dessen, was in dem Raum ist und wie du vorhast, die Monster aufzuspüren und zu fangen.

_____

_____

_____

_____

_____

_____

Name_____

2. Was sind die verschiedenen Teile, in die du dein Projekt zerlegt hast, um dir beim Design zu helfen? Denke daran, dass du mindestens drei Teile haben solltest: den Testraum, den physischen Monsterjäger (um die Monster mit Edison aufzuspüren und zu fangen) und das Programmdesign.

_____

_____

_____

_____

_____

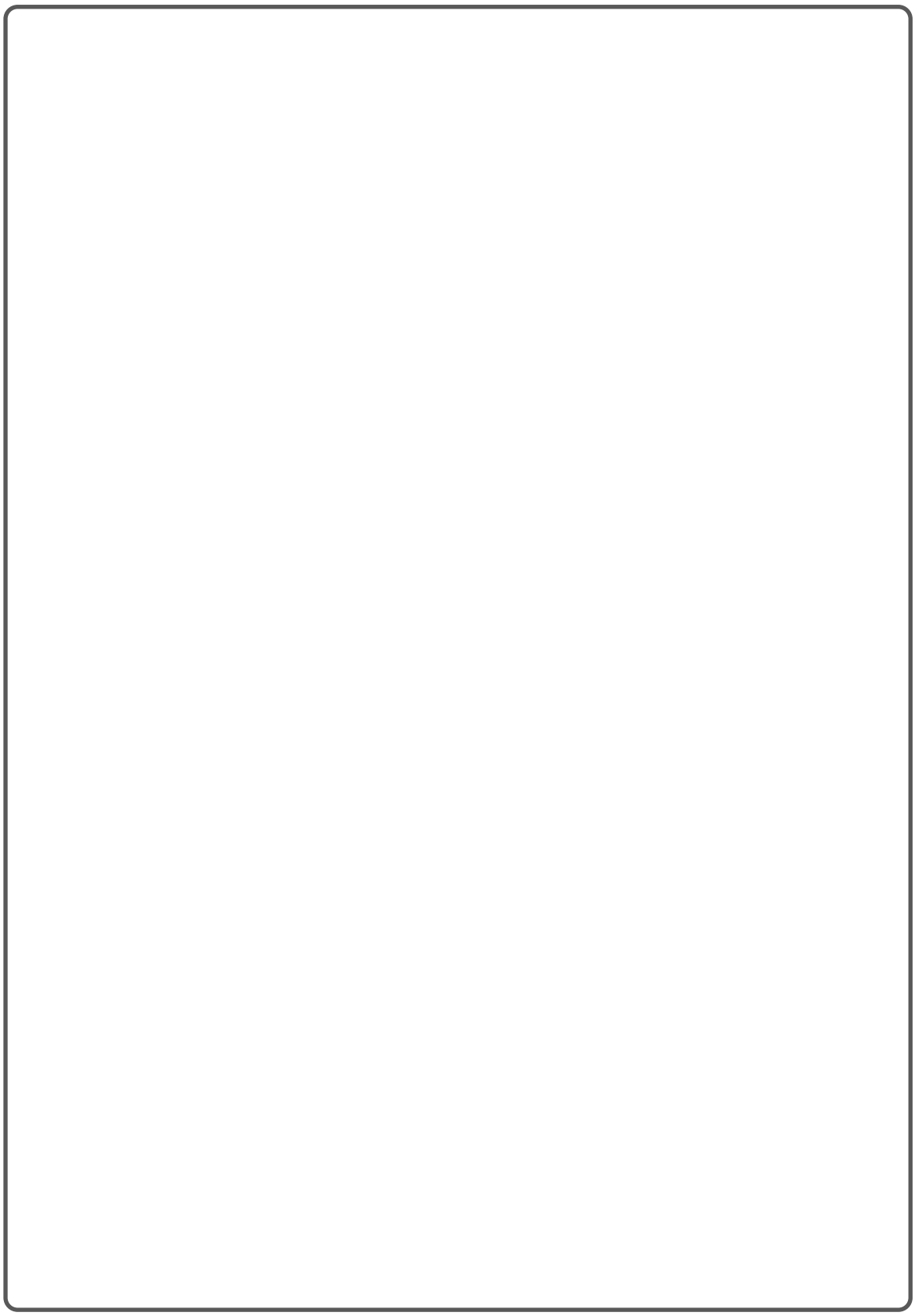
3. Gestalte dein Zimmer (Testraum). Benutze extra Platz, wenn du ihn brauchst.

Name_____

4. Gestalte deinen Monsterjäger (Edison-Erschaffung). Benutze extra Platz, wenn du ihn brauchst.

Name_____

5. Entwerfe dein EdScratch-Programm für deine Kreation. Benutze extra Platz, wenn du ihn brauchst.



Name_____

Raum 2

Beantworte die Fragen und nutze diesen Raum, um den Raum, die Monster und deine Falle zu gestalten. Sobald du deinen Entwurf für jeden Teil hast, baue deinen Raum, die Monster und deine Monsterjagd-Erfindung. Programmiere deine Erfindung in EdScratch, dann teste deine Kreation in deinem Raum, um zu sehen, was funktioniert und was verbessert werden muss.

Wende das, was du aus deinem Test gelernt hast, auf deinen Entwurf an, erstelle die nächste Iteration deines Builds und teste erneut. Wiederholt den Design-Build-Test-Zyklus so lange, bis ihr in der Lage seid, alle Monster aufzuspüren und einzufangen.

6. Welche Monster sind in diesem Raum? Wie viele Monster gibt es hier? Schreibe eine kurze Beschreibung dessen, was in dem Raum ist und wie du die Monster aufspüren und fangen willst.

_____

_____

_____

_____

_____

_____

7. Was sind die verschiedenen Teile, in die du dein Projekt zerlegt hast, um dir beim Design zu helfen? Denke daran, dass du mindestens drei Teile haben solltest: den Testraum, den physischen Monsterjäger (um die Monster mit Edison aufzuspüren und zu fangen) und das Programmdesign.

_____

_____

_____

_____

_____

Name_____

8. Gestalte dein Zimmer (Testraum). Benutze extra Platz, wenn du ihn brauchst.

Name_____

9. Gestalte deinen Monsterjäger (Edison-Erschaffung). Benutze extra Platz, wenn du ihn brauchst.

Name_____

10. Entwerfe dein EdScratch-Programm für deine Kreation. Benutze extra Platz, wenn du ihn brauchst.

Name_____

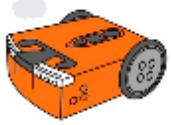
Arbeitsblatt U6-1: Sechs Ideen

1

2

3

4

5

6