Leitprogramm Kara

Ein Einstieg ins Programmieren mit Automaten

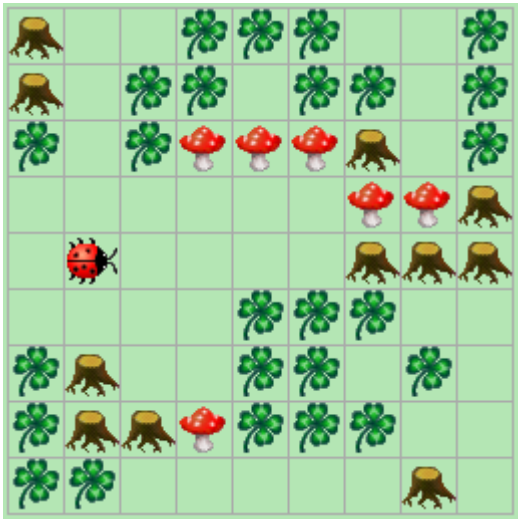
Schulstufe: Gymnasium
Zeitaufwand: ca. 8 Stunden
Autorin: Judith Zimmermann
Betreuer: Juraj Hromkovic
Fassung: 6.5.05
Keine Schulerprobung

Die Kara-Umgebung und alle Aufgaben, die Kara in diesem Leitprogramm gestellt werden, wurden von Raimond Reichert entwickelt. Auch die Beispiele von alltäglichen Automaten stammen von Raimond Reichert.

Ich bedanke mich bei Martina Zimmermann und Raimond Reichert und für die hilfreichen Hinweise.

Einführung

Kara ist ein Marienkäfer. Seine Welt ist eine Wiese, die in quadratische Felder aufgeteilt ist. Kara kann von Feld zu Feld über die ganze Wiese gehen. Auf den einzelnen Feldern können Kleeblätter und Pilze wachsen oder alte Baumstrünke herum liegen. Kara soll mit unserer Hilfe verschiedene Aufträge erledigen können. So soll er zum Beispiel alle vor ihm liegenden Blätter einsammeln.



Deine Aufgabe ist es nun Kara so zu **programmieren**, dass er **selbständig seine Aufträge erfüllen kann**. Damit du Kara geschickt ans Ziel steuern kannst, kennt er einige **Befehle**, die er folgsam ausführt. Zum Beispiel kannst du Kara die Anweisung geben, einen Schritt nach vorne zu gehen. Aber wenn Kara einen Schritt nach vorne macht, kann es sein, dass vor ihm ein Baumstrunk liegt und er sich den Kopf anstösst. Damit dies nicht passiert, versteht Kara einige **Fragen**, die er mit **ja oder nein beantworten** kann. So kannst du ihn zum Beispiel fragen, ob ein Baum auf dem Feld vor ihm liegt. Falls „nein“, kannst du ihm die Anweisung geben einen Schritt nach vorne zu tun. Falls ein Baum vor Kara liegt, so musst du ihn darum herumführen. Zum Beantworten der Fragen benutzt Kara **Sensoren**. Mit Hilfe der Sensoren kann Kara zum Beispiel feststellen, ob auf dem Feld vor ihm ein Baumstrunk steht oder nicht, ohne dass er selber das Feld begehen muss.

Kara programmierst du mit Hilfe von **„Wenn..., dann...“-Aussagen**. Das sieht zum Beispiel wie folgt aus: **Wenn** Kara nicht vor einem Baum steht, **dann** soll er einen Schritt nach vorne gehen. Oder: **Wenn** Kara auf einem Blatt steht, **dann** soll er es aufnehmen. Eine **Folge** von solchen **„Wenn..., dann...“-Anweisungen** ergibt ein **Programm**. Anschliessend kannst du Kara mit deinem Programm starten und seinen Lauf über die Wiese am Bildschirm mitverfolgen. Verhält sich Kara in jeder Situation so wie du es dir gedacht hast? Kann er die Aufgabe lösen?

Das gleiche Prinzip mit dem Kara programmiert wird, steckt auch in vielen dir bekannten Geräten: Videospieler, Kaffeemaschine, Getränkeautomat und vielen mehr.

Was kannst du nach der Bearbeitung dieses Leitprogramms:

- Du hast eine Vorstellung dafür entwickelt, was ein Programm ist, wie es abläuft, und wie es auf Einflüsse der Umwelt reagieren kann. Das erleichtert dir den Umgang mit bestehender Software.
- Beim Programmieren schärfst du dein Verständnis für Logik. Nur durch korrektes Erkennen der Situation und geschicktes Anleiten von Kara, kann er die Aufgaben lösen.

Inhaltsverzeichnis

Arbeitsanleitung

Mit Hilfe dieses Leitprogramms lernst du, Kara zu programmieren. Du bearbeitest jeweils ein Kapitel selbständig und meldest dich anschliessend beim Lehrer oder der Lehrerin für den Kapiteltest. Die einzelnen Kapitel sind immer gleich strukturiert:

Übersicht Was wird in diesem Kapitel behandelt? Was gibt es zu tun?

Lernziele Was kann ich nach dem Bearbeiten des Kapitels?

Im Teil mit dem eigentlichen Lernstoff triffst du auf die folgenden Zeichen:

Definition

Information

Wissenssicherung

Aufgabe

Lernkontrolle Habe ich alles verstanden?

Lösungen zu allen gestellten Problemen.

Nach der Lernkontrolle kannst du dich bei deiner Lehrerin oder deinem Lehrer für den Kapiteltest melden.

Bei Unklarheiten...

Solltest du einen Abschnitt auch nach gründlichem Durchlesen nicht verstehen, so fragst du eine Kameradin oder einen Kameraden. Falls ihr auch zu zweit nicht weiter kommt, dürft ihr gerne den Lehrer oder die Lehrerin fragen.

Muss das ganze Leitprogramm bearbeitet werden?

Nein. Kapitel 1, 2 und 3 sind obligatorisch. Kapitel 4 ist eine Aufgabensammlung. Die einzelnen Aufgaben haben jeweils einen Schwierigkeitsgrad zwischen 1 und 5, wobei 5 am schwierigsten ist. Wenn du Zeit hast, kannst du nach Lust und Laune einzelne Aufgaben herauspicken und lösen.

Kapitel 1 Kara und seine Umgebung

Übersicht

Worum geht es?

In diesem Kapitel wirst du Kara und seine Umgebung kennen lernen. Welche Befehle kennt Kara? Was trifft er auf seiner Wiese an und welche Eigenschaften haben diese Gegenstände? Um mit Karas Umgebung und Kara vertraut zu werden, wirst du ihn von Hand Schritt für Schritt steuern und so kleine Aufgaben lösen.

Was tust du?

Lies zunächst die Abschnitte der Reihe nach durch. Mache dich mit Karas Welt bekannt. Spiele ein bisschen damit herum. Du kannst durchaus auch mehr ausprobieren, als dir die Aufgaben vorgeben.

Lernziele

Nach dem Bearbeiten dieses Kapitels,

- kennst du Kara und seine Umgebung.
- kannst du Kara eine Welt zusammenstellen.
- kannst du Kara von Hand steuern.
- weisst du was Sensoren sind.

1.1. Karas Welt

Starte zunächst das Programm von Kara mit einem Doppelklick der linken Maustaste auf das Kara-Programm. Wähle nun die Umgebung „Kara – programmieren mit Automaten“ (Gelb hervor gehoben). Es erscheint das folgende Fenster auf dem Bildschirm:

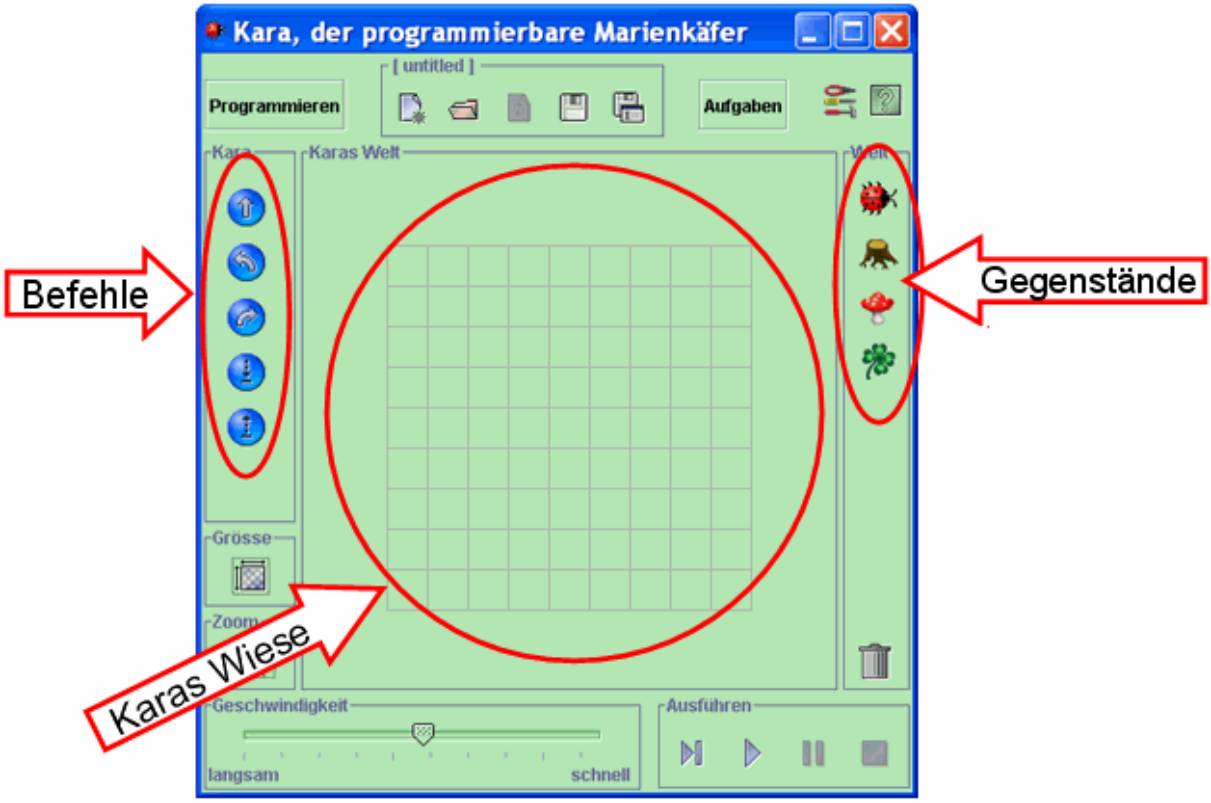


Abb. 1.1: Karas Welt. Im Zentrum ist Karas in Rechtecke aufgeteilte Wiese zu sehen. Rechts im Bild siehst du die Gegenstände, die in Karas Welt existieren. In jedem Viereck kann eines oder mehrere Gegenstände platziert werden. Links im Bild findest du alle Befehle, die Kara ausführen kann.

Gegenstände

Die Welt von Kara kannst du selber gestalten. Dafür platzierst du beliebig viele Gegenstände auf der Wiese.

Das ist Kara. Kara kann in der Welt immer nur einmal vorkommen.

Kara kann Kleeblätter aufnehmen und ablegen. Er schleppt immer einen grossen Sack voller Kleeblätter mit sich herum.

Der Baumstrunk beansprucht ein Feld für sich alleine. Kara kann nicht über einen Baumstunk krabbeln, sondern muss darum herum gehen.

Pilze können mit Kleeblättern zusammen auf einem Feld stehen. Kara kann nicht über Pilze krabbeln, aber er kann einen Pilz vor sich her schieben. Steht mehr als ein Pilz direkt vor ihm, so hat er nicht die Kraft die Pilze zu verschieben.

Ziehe mit Drag-and-Drop (Gegenstand auswählen und mit gedrückter Maustaste ans Ziel ziehen) der gewünscht Gegenstand auf die Wiese. Willst du einen Gegenstand löschen, so ziehe ihn mit Drag-and-Drop von der Wiese auf den Mülleimer.

Befehle

Kara kennt die Befehle auf der linken Seite des Feldes, und führt sie folgsam aus.


Kara hüpft auf das vor ihm liegende Feld. Falls ein Baum darauf steht, kann er nicht vorwärts gehen.


Kara bleibt auf dem Feld stehen und dreht sich nach links.


Kara bleibt auf dem Feld stehen und dreht sich nach rechts.

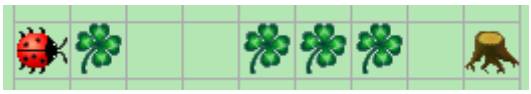

Kara legt ein Kleeblatt ab, falls das Feld leer ist. Liegt schon ein Kleeblatt auf dem Feld, so erscheint ein Hinweis.


Kara nimmt das Kleeblatt das auf dem Feld liegt auf. Falls kein Kleeblatt auf dem Feld liegt, so erscheint ebenfalls ein Hinweis.

In der folgenden Aufgabe, kannst du das Bedienen von Kara üben...

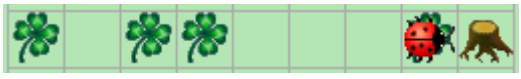

Wissenssicherung 1.1

Baue die folgende Welt für Kara auf:



Die Aufgabe von Kara ist nun zum Baumstrunk zu gehen. Unterwegs soll er jedes Kleeblatt aufnehmen. Liegt kein Kleeblatt auf einem Feld, so soll Kara eines hinlegen. Benutze dazu die Befehle am linken Rand.

Am Schluss soll es so aussehen:



Lass Kara nun noch einen Schritt geradeaus machen. Was passiert?

Sensoren

Wenn wir Kara von Hand steuern, dann können wir für ihn die Aufgabe des Sehens übernehmen. So überprüfst du in Aufgabe 1.1 in jedem Schritt, ob Kara auf einem Kleeblatt steht und gibst ihm dann den Befehl, ein Kleeblatt abzulegen oder eines aufzunehmen. Anschliessend hast du kontrolliert, ob Kara vor einem Baum steht, oder ob er noch einen Schritt weiter gehen kann.

Nehmen wir an, du müsstest Kara so programmieren, dass er eine bestimmte Aufgabe nur gerade in einer einzigen Welt auszuführen hat, dann kannst du voraussehen welchen Weg Kara gehen soll und nach wie vielen Schritten er jeweils ein Kleeblatt aufnehmen oder ablegen soll. Aber deine Aufgabe ist es, Kara so zu programmieren, dass er seinen Auftrag in verschiedenen Welten ausführen kann. So soll Kara zum Beispiel auf dem Weg zum Baum unterschiedlich viele Kleeblätter aufnehmen können, je nachdem wie viele dort liegen.

Damit du diese Aufgabe lösen kannst, kannst du Kara die unten aufgeführten Fragen stellen, die er dann mit Hilfe der Sensoren mit ja oder nein beantwortet. Du kannst Kara zum Beispiel fragen, ob er auf einem Kleeblatt steht oder nicht.

	Steht Kara vor einem Baumstrunk?
	Ist links von Kara ein Baumstrunk?
	Ist rechts von Kara ein Baumstrunk?
	Steht Kara vor einem Pilz?
	Steht Kara auf einem Kleeblatt?

Aufgabe 1.2

Welche Sensoren würdest du einsetzten, damit Kara den Auftrag aus Aufgabe 1.1 erledigen kann?

Aufgabe 1.3

Gegeben sei diese Welt:

Kara startet in der angezeigten Position und macht dann nacheinander die folgenden Aktionen:

Fülle die untenstehende Tabelle aus. Überlege dir was Karas Sensoren in jedem Schritt „sehen“. Falls als Antwort „nein“ zurückgegeben wird so mache ein × und falls ein „ja“ dann ein ✔ .

	Start	nach 1 Schritt	nach 2 Schritten	nach 3 Schritten	nach 4 Schritten	nach 5 Schritten
 Baum vorne?	×					
 Baum links?	✔					
 Baum rechts?	×					
 Pilz vorne?	×					
 Kleeblatt unten?	✔					

Kapiteltest

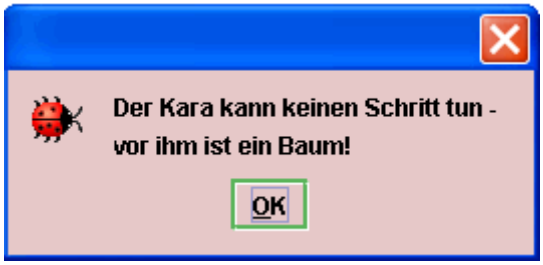
Wenn dir klar ist wie eine Welt für Kara zusammengebaut werden kann und wie Kara von Hand gesteuert wird, dann kannst du bei der Lehrerin oder dem Lehrer den Kapiteltest holen.

Lösungen Kapitel 1

Wissenssicherung 1.1

Diese Befehlsfolge bringt Kara an sein Ziel:

Macht Kara noch einen Schritt vorwärts, so läuft er in den Baumstrunk:



Aufgabe 1.2

Damit du Kara die richtigen Anweisungen geben kannst, musst du wissen, ob er auf einem Kleeblatt steht und ob er schon beim Baum angekommen ist. Also brauchst du die folgenden beiden Sensoren:

Aufgabe 1.3

	Start	nach 1 Schritt	nach 2 Schritten	nach 3 Schritten	nach 4 Schritten	nach 5 Schritten
Baum vorne?	×	×	×	×	×	✔
Baum links?	✔	×	×	×	×	×
Baum rechts?	×	×	✔	×	×	×
Pilz vorne?	×	×	×	×	×	×
Kleeblatt unten?	✔	×	×	✔	×	✔

Kapitel 2 Alltägliche Automaten

Übersicht

Worum geht es?

In diesem Kapitel wirst du **Automaten** anhand von einigen alltäglichen Beispielen kennen lernen. Unter Automaten verstehen wir hier nicht das Gleiche wie in der Alltagsprache. Als Automat wird die **Steuerung** eines Getränkeautomaten oder eines Geldautomaten bezeichnet. Das gleiche Prinzip wirst du im nächsten Kapitel zum Programmieren von Kara verwenden.

Automaten basieren auf „**Wenn**..., **dann**...“-**Aussagen**. **Wenn** gewisse **Bedingungen** erfüllt sind, **dann** sollen bestimmte **Aktionen** ausgeführt werden. Wenn zum Beispiel bei einem Getränkeautomaten genügend Geld eingeworfen wurde und Cola gewählt wurde, dann soll der Getränkeautomat eine Cola Dose ausspucken.
Automaten haben auch ein Gedächtnis. Der Getränkeautomat weiss zum Beispiel, wie viel Geld bereits eingeworfen wurde.

Was tust du?

Bearbeite Abschnitt für Abschnitt. Kommst du an eine Aufgabe, so versuche sie zu lösen. Verschiebe das Lösen der Aufgabe nicht auf den Schluss. Im weiteren Verlauf wird davon ausgegangen, dass du die Aufgaben gelöst hast. Lege dir gleich einige Blätter Entwurfspapier bereit. Den Computer brauchen wir in diesem Kapitel nicht.

Lernziele

Nach dem Bearbeiten dieses Kapitels

- weisst du **wie ein Automat funktioniert**.
- kennst du die **fünf** Begriffe: **Zustand**, **Übergang**, **Input**, **Aktionen**, **Startzustand**.

2.1. Was Kara und Yoga gemeinsam haben

Kara wird mit Hilfe von Automaten programmiert. Wir werden zuerst einen kleinen Abstecher in eine Yoga-Stunde machen und dabei Automaten kennen lernen.

Beim Yoga nimmt man eine spezielle Stellung ein, zum Beispiel die Grussposition (Position 1) in Abbildung 2.1. Man bleibt für einige Zeit in dieser Position, bis der Lehrer oder die Lehrerin eine Anweisung gibt und wechselt dann in die nächste Stellung. In der neuen Stellung wartet man die nächste Anweisung ab und so weiter und so fort.

In das „Wenn..., dann...“- Schema übertragen würde das heisse: **Wenn** der Yogi in der Position 1 steht und der Lehrer die Anweisung gibt die Arme über den Kopf zu nehmen, **dann** soll der Yogi in Position 2 wechseln.

Bei der Wahl der nächsten Instruktion, muss die Lehrperson die momentane Körperhaltung des Yogi beachten. Steht der Yogi zum Beispiel schon auf einem Bein und bekommt die Anweisung ein Bein vom Boden zu heben, so landet er unsanft auf dem Boden.

Ein Ablauf in Yoga kann wie folgt aussehen:

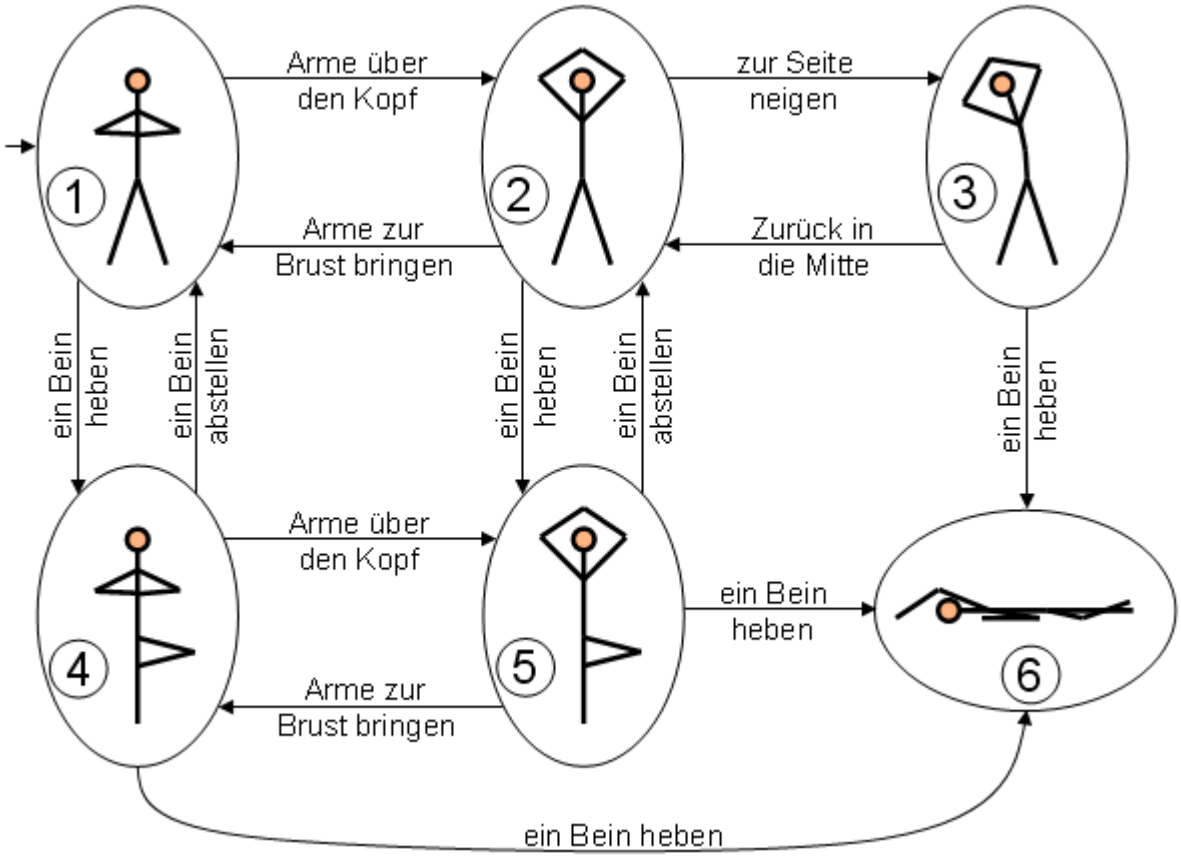


Abb. 2.1: Eine Yogastunde.

Eine Yogastunde wie in Abbildung 2.1 und ein Automat haben vieles gemeinsam.

Automat	Yogalektion
Ein Automat bleibt in einem Zustand und wartet darauf über einen Sensor von aussen einen ...	Der Yogi bleibt in der gleichen Position und wartet die nächste ...
Input zu erhalten. Angestossen durch diesen Input wählt er den ...	Anweisung der Lehrperson ab. Sagt der Guru zum Beispiel „Arme über den Kopf nehmen“, so folgt der Yogi dem ...
Übergang, der mit dem gegebenen Input beschriftet ist, führt die angegebenen ...	Pfeil mit der Beschriftung „Arme über den Kopf nehmen“. Dann ...
Aktionen aus und wechselt entweder in einen anderen Zustand oder bleibt im alten Zustand.	führt er die Anweisung des Lehrers aus, das heisst, der Yogi hebt die Arme in die Höhe und ist dann in der Position am anderen Ende des Pfeils.
Jeder Automat hat einen Startzustand, in dem er mit der Arbeit beginnt.	Damit die Lernenden auch mit der Yogastunde beginnen können, wenn der Guru zu spät kommt, beginnen sie immer in der Grussposition.

Im nächsten Abschnitt werden wir einen ersten Automaten aus dem Alltag etwas genauer betrachten.

2.2. Getränkeautomaten

Sicher hast du schon einmal an einem Getränkeautomat eine Dose gekauft. Wir wollen nun überlegen, wie wir einen Getränkeautomat steuern können.

Abb. 2.2: Getränkeautomat. Die Durstigen werfen Geld in den Automaten und wählen das gewünschte Getränk. Der Automat gibt auf der Kreditanzeige den schon eingeworfenen Betrag an. Wenn ein Getränk gewählt wurde und genügend Kredit vorhanden ist, so spuckt er die gewünschte Dose aus und setzt die Kreditanzeige auf Null.

Der Getränkeautomat soll wie folgt funktionieren:

- er akzeptiert nur 1- und 2-Fränkler
- er zeigt den schon eingeworfenen Betrag an
- er gibt nur Cola und ein weiteres Getränk aus
- eine Dose kostet 3.- Franken

Nun wollen wir uns das Innenleben des Automaten etwas genauer betrachten.

Der Automat muss 4 verschiedene Situationen unterscheiden können: 0,1,2 und 3 Franken Kredit. Dies macht er mit Hilfe von 4 Zuständen. Jeder Zustand steht für eines der möglichen Guthaben.

Abb. 2.2: Die vier Zustände des Getränkeautomaten.

Die Kunden könnten natürlich auch mehr Geld einwerfen. Aber der Einfachheit halber nehmen wir an, dass die Kunden maximal 3 Franken einwerfen.

Als nächster Schritt fügen wir die Übergänge hinzu, das heisst die Pfeile von einem Zustand zum nächsten. Auf den Übergängen notieren wir zuerst den benötigten Input (z.B.: Die Kundschaft wirft 2.- ein.) und anschliessend nach einem Schrägstrich die vom Automaten auszuführenden Aktionen (z.B.: Anzeige von 2 Franken auf dem Display).

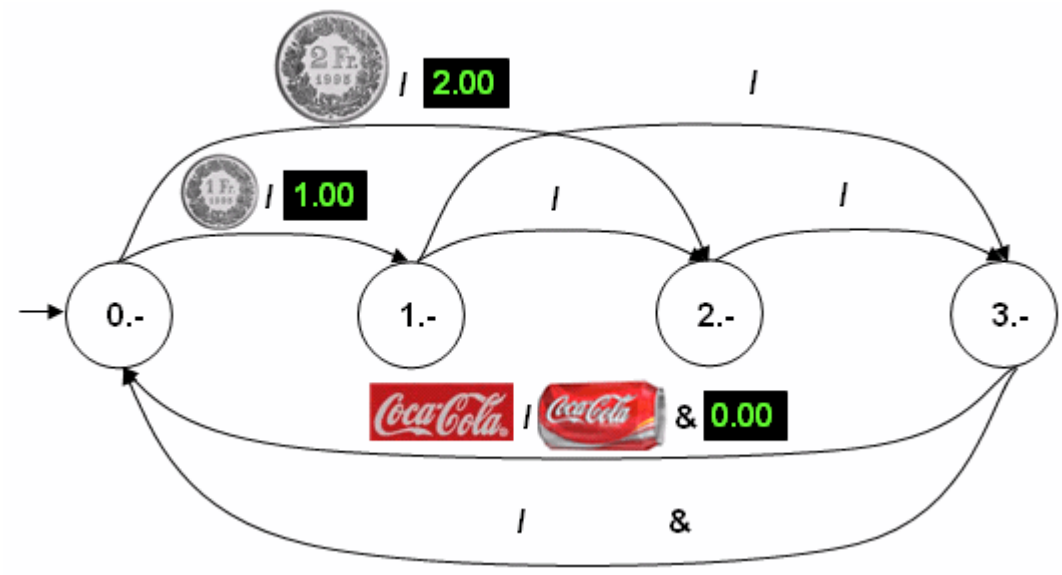


Abb. 2.3: Getränkeautomat mit Übergängen. In diesem Automat sind alle für uns interessanten Übergänge eingetragen. Drei der sieben Übergänge sind schon vollständig beschriftet.
Betrachten wir zunächst den Zustand „0.- Franken Kredit“ (Zustand ganz links). Es gibt zwei Übergänge aus diesem Zustand. Der Übergang in den Zustand „1.- Franken Kredit“ darf oder muss genommen werden, wenn ein 1-Fränkler eingeworfen wird. Zusätzlich muss auf dem Display „1.00“ angezeigt werden. Wird ein 2-Fränkler eingeworfen, so wird der andere Übergang genommen und in den Zustand „2.- Franken Kredit“ gewechselt. Auf dem Display soll „2.00“ angezeigt werden.
Der dritte, beschriftete Übergang führt aus dem Zustand „3.- Franken Kredit“ in den Zustand „0.- Franken Kredit“. Die Bedingung ist, dass die Durstigen den Cola-Knopf drücken. Dann gibt der Automat eine Cole-Dose aus und setzt das Display auf „0.00“.

Wissenssicherung 2.1

a) Beim Automat in der Abbildung 2.3 sind nicht alle Übergänge beschriftet. Vervollständige das Schema des Getränkeautomaten. Du kannst selber wählen, was für ein weiteres Getränk dieser Automat verkauft.
b) Welcher Zustand ist der Startzustand?
c) Mache 3 „Wenn..., dann...“-Aussagen zum Getränkeautomaten.

Aufgabe 2.2

Zeichne einen Automaten, der einen Lichtschalter steuert. Ein Sensor meldet dem Automaten, wenn der Schalter gedrückt wurde.

Die Aufgabe ist dann gut gelöst, wenn:
- der Automat korrekt arbeitet.
- die Zustände sprechende Namen haben.
- alle Übergänge eingezeichnet und mit Input und Aktionen beschriftet sind.

Aufgabe 2.3

Zeichne einen Automaten für die Steuerung eines Videorecorders. Der Videorecorder hat der Einfachheit halber nur die folgenden Tasten: ▶ ▶▶ ■.

Bei dieser Aufgabe sollen alle möglichen Übergänge eingezeichnet werden. Das heisst, dass es aus einem Zustand für alle möglichen Inputs einen Übergang geben muss. Unter Umständen führt der Übergang wieder in denselben Zustand zurück. Bisher haben wir wegen der besseren Übersichtlichkeit jeweils nur die wichtigen Übergänge eingezeichnet.

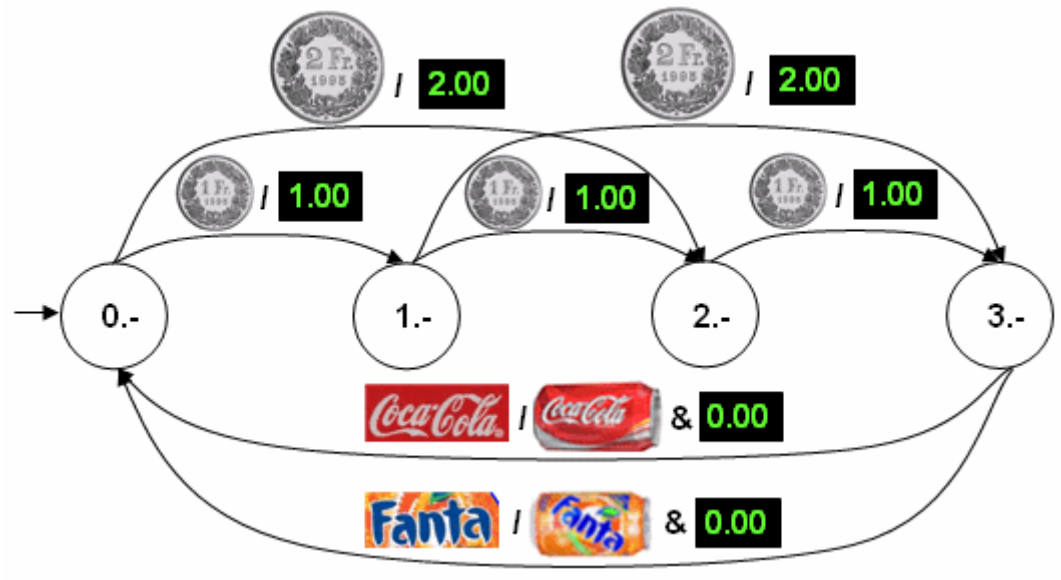
Die Aufgabe ist dann gut gelöst, wenn:
- der Automat korrekt arbeitet.
- die Zustände sprechende Namen haben.
- alle Übergänge eingezeichnet und mit Input und Aktionen beschriftet sind.
- der Startzustand angegeben ist.

Kapiteltest

Wenn die Begriffe Zustand, Übergang, Input, Aktionen und Startzustand klar sind, so kannst du zur Lehrperson gehen und den Kapiteltest holen.

Lösungen Kapitel 2

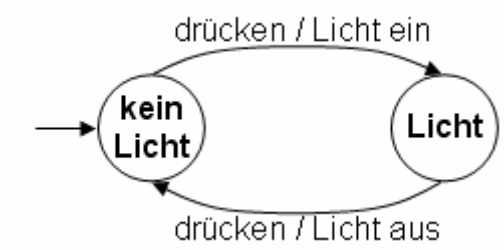
Wissenssicherung 2.1
a) + b) Als zweites Getränk wurde Fanta gewählt.



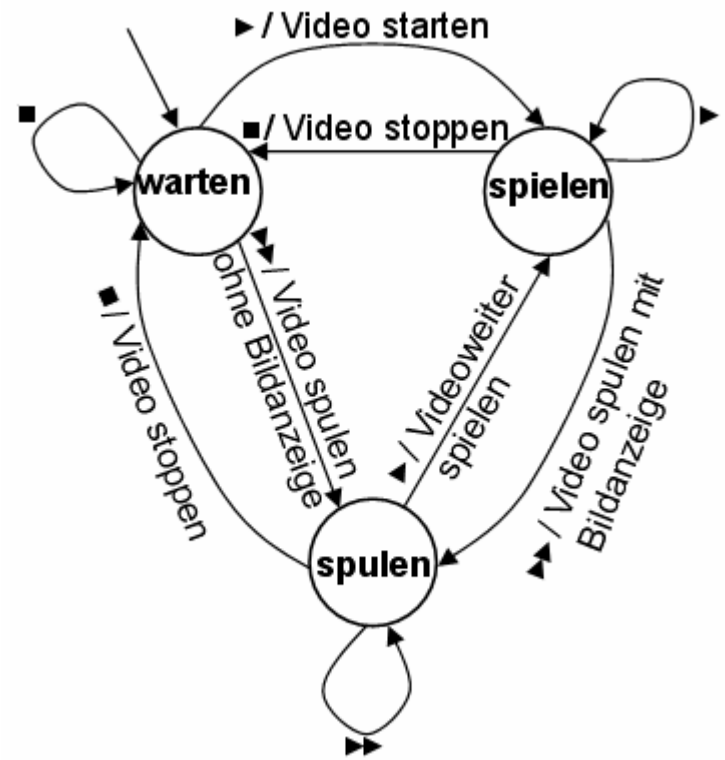
c) Es gibt sehr viele Möglichkeiten. Hier ein paar Beispiele:

- Wenn der Automat im Zustand „0.- Franken Kredit“ ist und jemand einen 1-Fränkler einwirft, dann soll der Automat „1.00“ auf dem Display anzeigen und in den „1.- Franken Kredit“ wechseln.
- Wenn der Automat im Zustand „1.- Franken Kredit“ ist und jemand einen 2-Fränkler einwirft, dann soll der Automat „3.00“ auf dem Display anzeigen und in den „3.- Franken Kredit“ wechseln.
- Wenn der Automat im Zustand „3.- Franken Kredit“ ist und auf den Cola-Knopf gedrückt wird, dann soll der Automat eine Cola-Dose ausspucken, „0.00“ auf dem Display anzeigen und in den Zustand „0.- Franken Kredit“ wechseln.

Aufgabe 2.2



Aufgabe 2.3



Es gibt auch andere Möglichkeiten, diese Aufgabe zu lösen. Wenn du dir nicht sicher bist, ob deine Lösung gut ist, besprich sie mit deinem Lehrer oder deiner Lehrerin.

Kapitel 3 Kara programmieren

Übersicht

Worum geht es?

In diesem Kapitel wirst du lernen, wie Kara mit Hilfe eines Automaten programmiert wird. Es ist wichtig, dass Kara seine Aufgabe jeweils in unterschiedlichen, aber gleich strukturierten Welten erfüllen kann. Kara kann zum Beispiel alle vor ihm liegenden Blätter aufnehmen, egal wie viele Blätter es sind und wie sie auf den Feldern verteilt sind. Die Welten bei den Aufgaben sind jeweils nur Beispiele.

Was tust du?

Lies wiederum die Abschnitte der Reihe nach durch und bearbeite Aufgaben direkt, wenn sie im Abschnitt auftauchen. Damit du dein Programm testen kannst, sind jeweils mehrere Welten vorgegeben. Du kannst natürlich auch selber noch zusätzliche Welten aufbauen. Lasse dein Programm in allen Welten laufen und überprüfe, ob Kara seine Aufgabe erledigen kann.

Lernziele

Nach dem Bearbeiten dieses Kapitels,

- kannst du Kara so programmieren, dass er einfache Aufgaben selbständig lösen kann.
- weisst du, wie du Fehler im Programm finden und beheben kannst.

Nun wollen wir Kara programmieren. Dafür schreiben wir für jede Aufgabe einen Automaten, der Kara durch seine Welt steuert. **Der Automat ist Karas Gehirn**. Er bekommt von den Sensoren **Informationen** über die Umgebung. Das ist der **Input** für unseren Automaten. Die **Aktionen** unseres Automaten sind die **Befehle an Kara**, die er ausführen soll.

Sensoren

	Steht Kara vor einem Baumstrunk?
	Ist links von Kara ein Baumstrunk?
	Ist rechts von Kara ein Baumstrunk?
	Steht Kara vor einem Pilz?
	Steht Kara auf einem Kleeblatt?

Befehle

 Kara hüpft auf das vor ihm liegende Feld. Falls ein Baum darauf steht, kann er nicht vorwärts gehen.

 Kara bleibt auf dem Feld stehen und dreht sich nach links.

 Kara bleibt auf dem Feld stehen und dreht sich nach rechts.

 Kara legt ein Kleeblatt ab, falls das Feld leer ist. Liegt schon ein Kleeblatt auf dem Feld so erscheint ein Hinweis.

 Kara nimmt das Kleeblatt das auf dem Feld liegt auf. Falls kein Kleeblatt auf dem Feld liegt, so erscheint ebenfalls ein Hinweis.

Betrachten wir ein einfaches Beispiel.

3.1. Kara auf Papier

Kara, die Blättersammlerin

Aufgabe: Schreibe ein Programm, das Kara geradeaus bis zum nächsten Baumstrunk führt. Kara weiss, dass geradeaus vor ihm ein Baumstrunk steht. Liegt auf einem Feld ein Blatt, soll Kara es aufnehmen. Beim Baumstrunk angekommen ist das Programm zu beenden. (Achtung: Nur wenn ein Baumstrunk vor Kara liegt, läuft das Programm korrekt.)

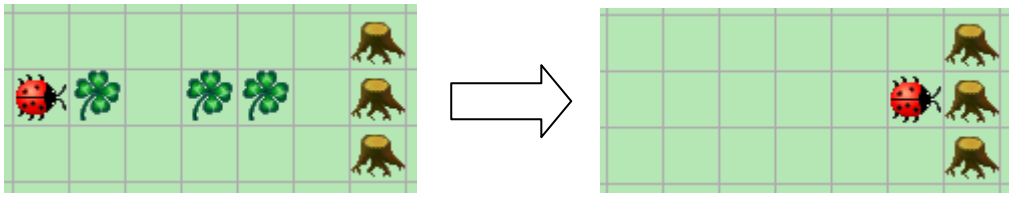


Abb. 3.1: Das Bild links zeigt die Ausgangslage und auf dem Bild rechts siehst du wie es aussehen muss, nachdem Kara ihren Auftrag erledigt hat.

Nun wollen wir zuerst den Automaten auf einem Blatt Papier entwerfen. Zunächst müssen wir uns überlegen, **welche Sensoren wir benötigen**. Zum Lösen dieser Aufgabe interessiert es uns nicht, ob Bäume links oder rechts von Kara sind. Wir müssen nur wissen, wann **Kara beim Baum angekommen** ist und ob **Kara auf einem Kleeblatt steht** oder nicht.

Welche Zustände kommen vor?

Wir brauchen zwei Zustände: einen Zustand zum Sammeln der Kleeblätter und den Stoppzustand. Der Stoppzustand soll erreicht werden, sobald Kara vor dem Baum steht. Unseren bisherigen Automaten haben die Arbeit nie beendet, sondern einfach auf neue Benutzende gewartet. Bei Kara ist es jedoch anders, wir wollen, dass Kara sagt, wenn er mit der Arbeit fertig ist. Das erreichen wir dadurch, dass er nach getaner Arbeit in den Stoppzustand wechselt.

Abb. 3.2: Die beiden Zustände des Automaten für die Blättersammlerin. So lange Kara mit der Arbeit nicht fertig ist, soll er im Zustand „Blätter sammeln“ bleiben. Sobald die Arbeit verrichtet ist, soll er in den Stoppzustand wechseln.

Welche Übergänge werden benötigt?

Als nächstes werden für alle möglichen Kombinationen der gewählten Sensoren einen Übergang hinzugefügt. Eine möglicher Input durch die zwei Sensoren ist: Vor Kara steht kein Baum und Kara steht nicht auf einem Kleeblatt. Dies entspricht der ersten Zeile in Abb. 3.2.

	
×	×
×	✔
✔	×
✔	✔

Abb. 3.3: In dieser Tabelle sind alle möglichen Kombinationen der Inputs aufgelistet, die Kara über die gewählten Sensoren erhalten kann. Ein Kreuz bedeutet, dass der Sensor „nein“ zurück meldet. Ein Häklein zeigt an, dass der Sensor „ja“ meldet.

Nun müssen wir uns noch überlegen, wie Kara in jedem der vier Fälle reagieren soll und in welchem Zustand er anschliessend sein muss. Solange die Aufgabe nicht gelöst ist, darf noch nicht in den Stoppzustand gewechselt werden. Sobald aber der Auftrag erledigt ist, muss in den Stoppzustand übergegangen werden.

		Karas Aktionen	nächster Zustand
×	×		Blätter sammeln
×	✔		Blätter sammeln
✔	×	-	Stopp
✔	✔		Stopp

Abb. 3.4: Diese Tabelle zeigt, welche Aktionen Kara aufgrund der Inputs der Sensoren ausführen soll. Die erste Zeile der Tabelle zeigt: Wenn Kara weder vor einem Baum, noch auf einem Kleeblatt steht, dann soll Kara einen Schritt nach vorne machen.

Die Zustände aus Abbildung 3.2 und die Übergänge aus Tabelle 3.3 ergeben den folgenden Automaten:

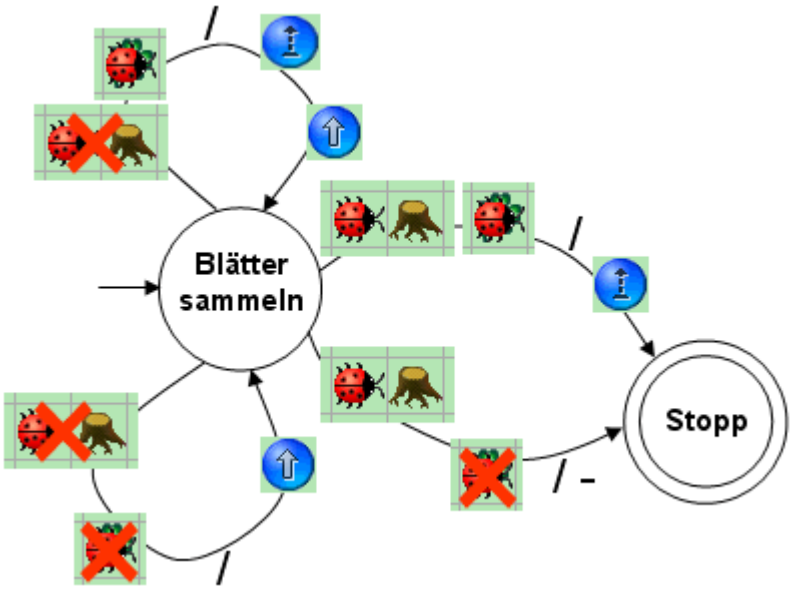
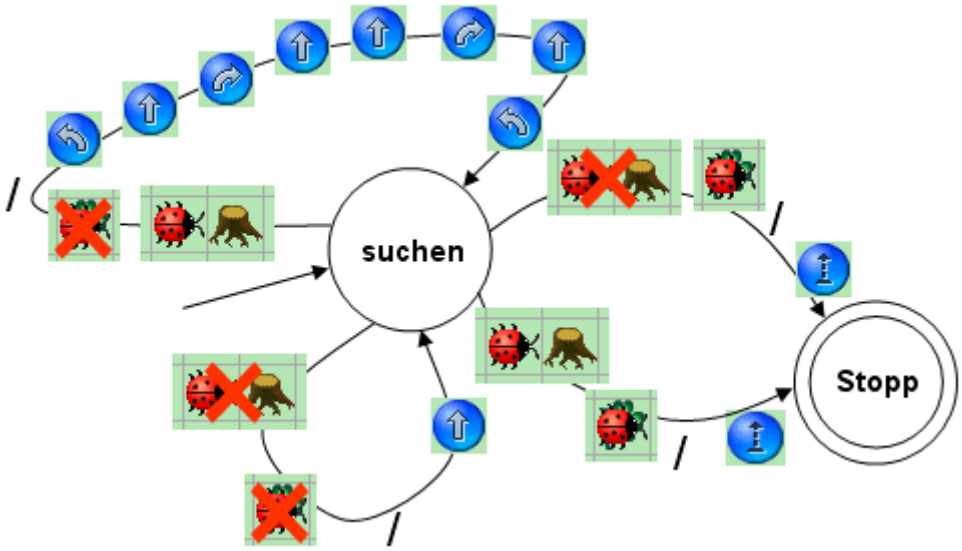


Abb. 3.5: Automat der Kara anleitet, geradeaus bis zum nächsten Baum zu gehen und unterwegs alle Blätter aufzunehmen

Wissenssicherung 3.1

In dieser Aufgabe geht es darum einen bestehenden Automaten zu analysieren.



Gegeben ist die unten stehende Wiese. Zeichne den Weg ein, der Kara geht, wenn er durch den oben abgebildeten Automaten gesteuert wird.

Hinweis: Wenn Kara während eines Ausweichmanövers in einen Baum rennt, dann wird eine Fehlermeldung ausgegeben und der Automat wechselt in den Stoppzustand.

Im folgenden Block werden wir Kara mit dem Automaten der Aufgabe 3.1 programmieren, so dass er selbstständig um die Baumstrünke kurven kann und das gesuchte Blatt im Wald findet.

3.2. Kara am Computer

Aufgabe 3.2

Auf den folgenden Seiten wirst du Schritt für Schritt angeleitet, den Automaten aus Aufgabe 3.1 zu programmieren. Lies die Abschnitte der Reihe nach durch und führe die mit dem Symbol ➤ gekennzeichneten Arbeitsschritte direkt aus.

Die Lösungen zu dieser Aufgabe sind jeweils gleich in den Text eingestreut. So kannst du immer kontrollieren, ob du richtig liegst. Die komplette Lösung findest du in den Lösungen.

Hinweis: Automat und Programm meinen bei Kara das gleiche.

➤ Starte Kara und wähle Kara- programmieren mit Automaten. Klicke dann oben links auf Programmieren. Nun wird das Programmierfenster für Kara geöffnet.

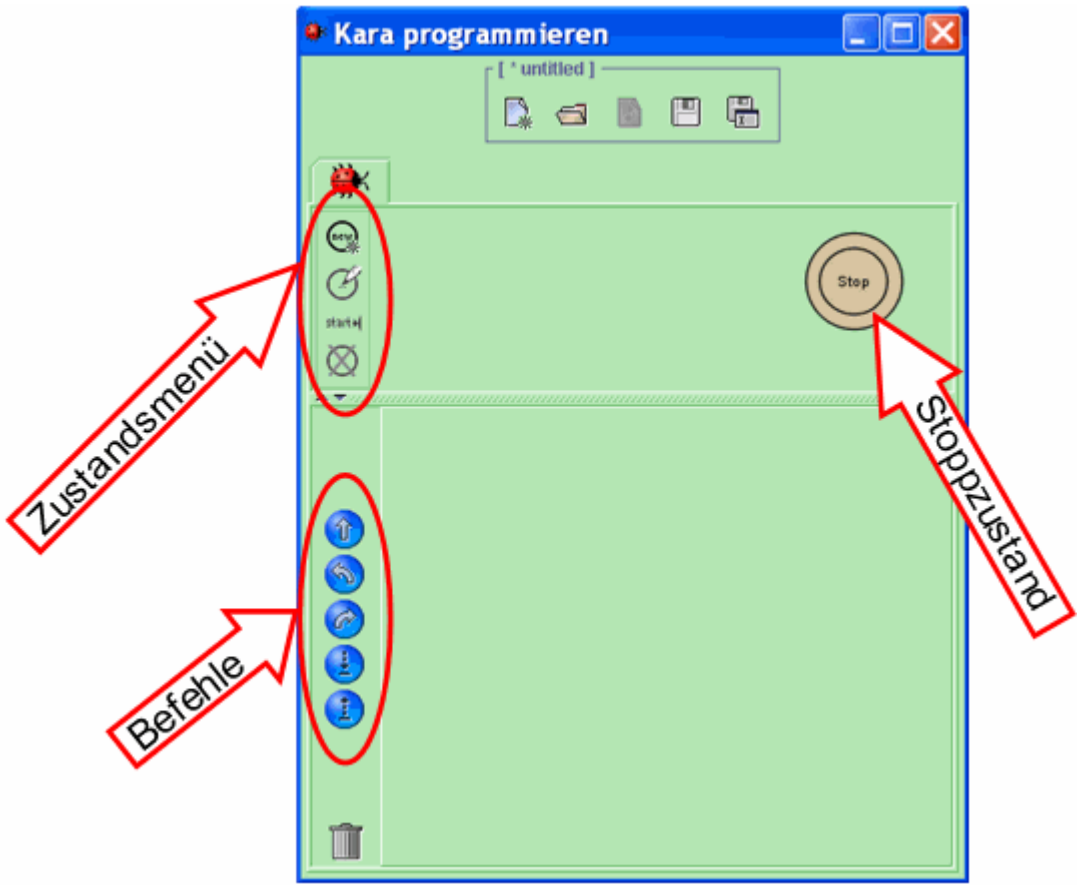


Abb. 3.6: Der Stoppzustand ist standardmässig schon gezeichnet. Im Feld, wo der Stoppzustand ist, kann der Automat gezeichnet werden. Links davon sind alle Operationen, die benötigt werden, um einen Automaten zu zeichnen.

Zustände einfügen

Die folgenden Operationen stehen dir zur Verfügung, um einen Automaten zu zeichnen:

	Um einen neuen Zustand zu erhalten, klickst du mit der Maus auf dieses Symbol. Es geht ein Zustandseditor auf, in dem du den Zustand editieren kannst.
	Einen Zustand kann man verändern, in dem man ihn zuerst mit der Maus anklickt und dann mit einem Mausklick auf dieses Symbol den Zustandseditor öffnet.
	Jeder Automat muss einen Startzustand haben. Markiere einen Zustand und klicke mit der Maus auf dieses Symbol, um den Zustand als Startzustand auszuzeichnen.
	Ein Mausklick auf dieses Symbol löscht den zuvor mit der Maus ausgewählten Zustand.
	Einen Übergang zwischen zwei Zuständen erhältst du, wenn du im Ausgangszustand in den äusseren Ring klickst, die Maustaste gedrückt hältst, den Mauszeiger in den Folgezustand ziehst und die Maustaste wieder los lässt.

- Zeiche einen neuen Zustand mit einem linken Mausklick auf das dieses Symbol  .

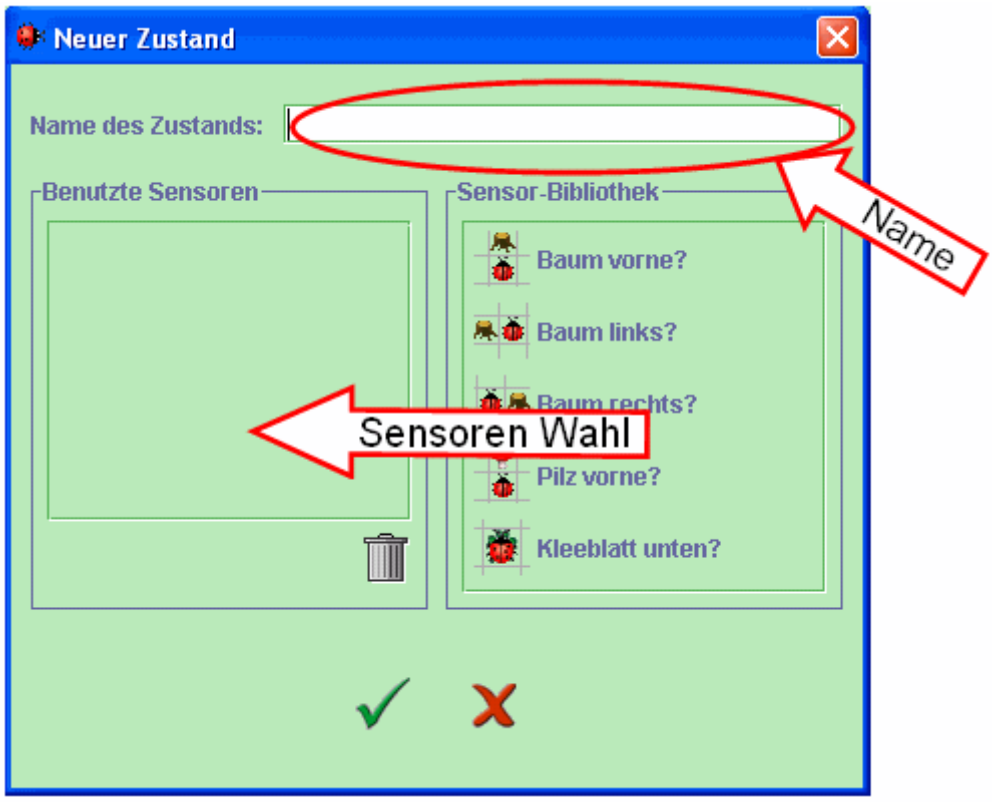


Abb. 3.7: Zustandseditor. Die gewählten Sensoren erscheinen im Übergangseditor des Zustandes. Der Übergangseditor liegt im Programmeditor unterhalb des Feldes, in dem der Automat gezeichnet wird (siehe Abbildung 3.8).

- Gib dem neuen Zustand im Namensfeld einen sprechenden Namen, zum Beispiel „suchen“.
- Wähle dann die benötigten Sensoren und ziehe sie mit Drag-and-Drop in den Rahmen unter „Benutzte Sensoren“. Für das Blättersammeln müssen wir wissen, ob Kara vor einem Baum steht, oder ob er auf einem Kleeblatt steht. Abbildung 3.8 zeigt dir, wo du die gewählten Sensoren im Programmeditor wieder findest.

Im Programmeditor ist nun ein zusätzlicher Zustand zu sehen. Es fehlen aber noch alle Übergänge. Diese werden wir im nächsten Schritt einfügen.

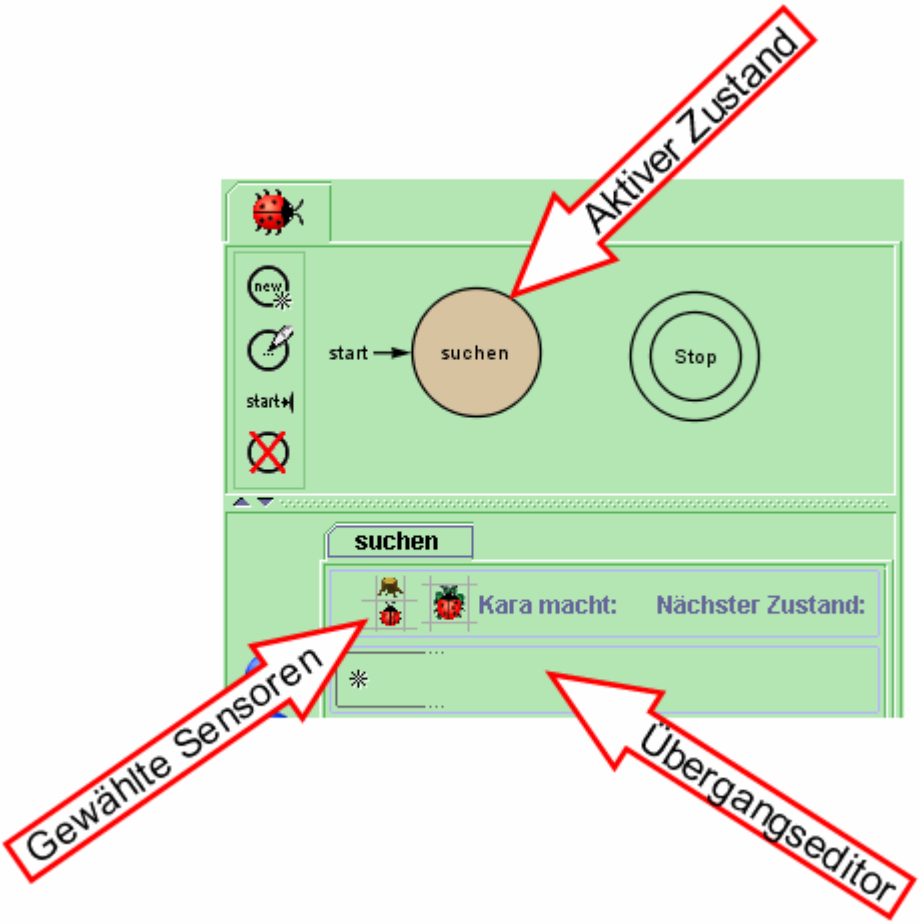


Abb. 3.8: Diese Abbildung zeigt den Programmeditor mit dem Übergangseditor für den aktiven Zustand. Im Übergangseditor werden die gewählten Sensoren angezeigt.

- Klicke zwischen dem „suchen“-Zustand und dem Stoppzustand hier und her. Wie verändert sich der Übergangseditor?

Hinweis: Im Stoppzustand benötigen wir keine Sensoren und Übergänge mehr, da die Arbeit abgeschlossen ist.

Übergänge

Die Übergänge werden wie folgt gezeichnet: Klicke in den äusseren Rand des Startzustandes und ziehe den Mauszeiger auf den Folgezustand, ohne die Maustaste loszulassen.

Hinweis: In der Ansicht des Automaten wird immer nur ein Übergang dargestellt. Im Übergangseditor kann die volle Anzahl Übergänge gesehen werden.

- Übernimm alle Übergänge des Automaten aus der Aufgabe Wissenssicherung 3.1.

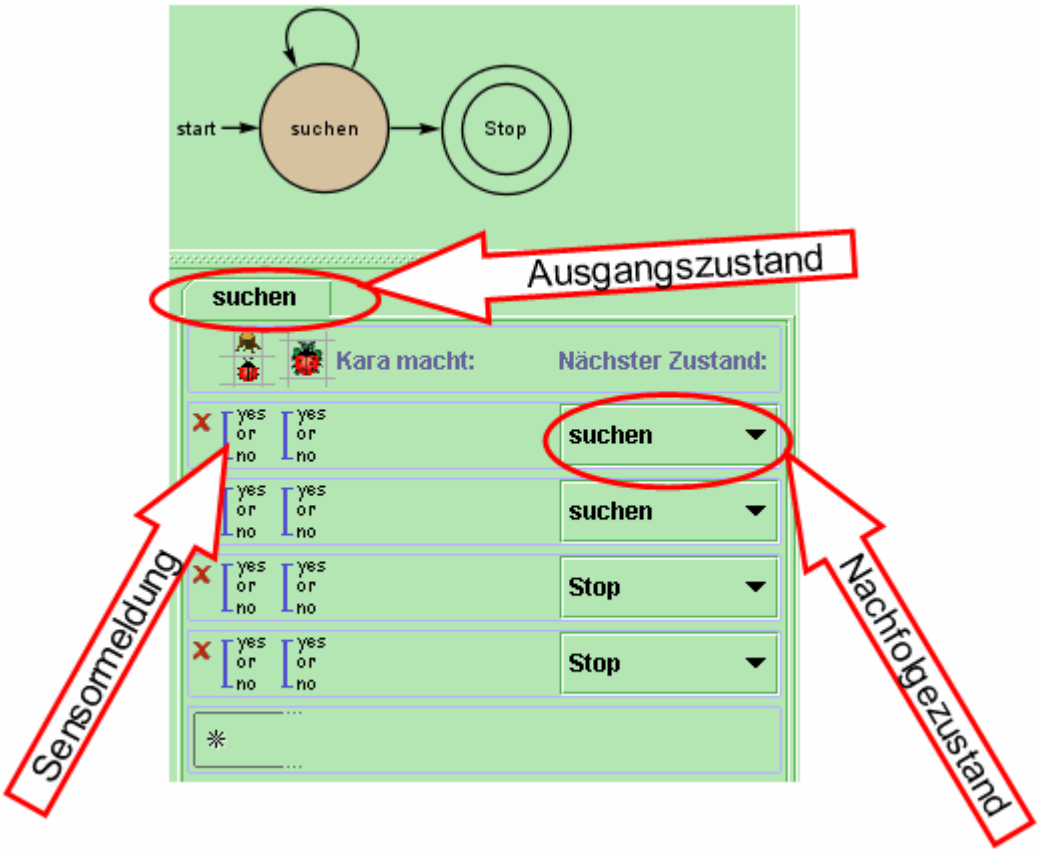


Abb. 3.9: Im unteren Teil des Programmeditors ist der Übergangseditor zu sehen. Hier können wir die Inputs (Bedingungen, Meldungen der Sensoren) und Aktionen (Anweisungen an Kara) der einzelnen Übergänge festlegen. Eine Zeile entspricht einem Übergang. Der Übergang geht vom Ausgangszustand in den Nachfolgezustand. Nachfolgezustände müssen nicht für alle Übergänge dieselben sein.

Nun fehlen nur noch die Inputangaben, wann ein Übergang genommen wird und die Aktionen, die darauf folgen müssen.

Input und Aktionen

Die Beschriftung der Übergänge mit Input und Aktionen machen wir im Übergangseditor. Soll Kara vor einem Baum stehen oder nicht, wenn er den Übergang in der ersten Zeile nimmt? Und soll er im Zustand „suchen“ bleiben? Mit einem Mausklick auf den blauen Balken kann die Belegung auf *yes* oder *no* geändert werden. Wenn es aber keine Rolle spielt, welche Meldung der Sensor gibt, so kann auch *yes or no* beibehalten werden. Anschliessend können die Befehle mit Drag-and-Drop in der gewünschten Reihenfolge rechts der Sensorbelegungen eingefügt werden.

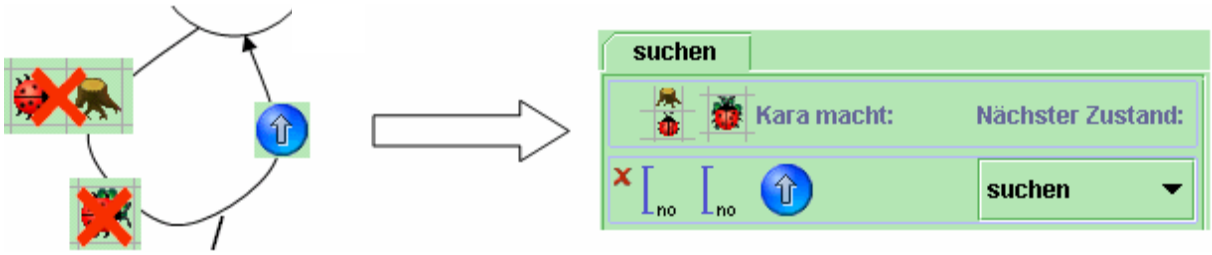


Abb. 3.10: In dieser Abbildung ist das Übertragen eines Übergangs vom Papier ins Kara Programm zu sehen.

- Füge alle Sensorbelegungen und die Anweisungen an Kara den Übergängen hinzu.

Nun ist der Automat fertig und du kannst Kara mit deinem Automaten laufen lassen. Dafür wechselst du wieder in das Fenster mit Karas Welt. Am unteren Rand findest du die folgenden Menüs:

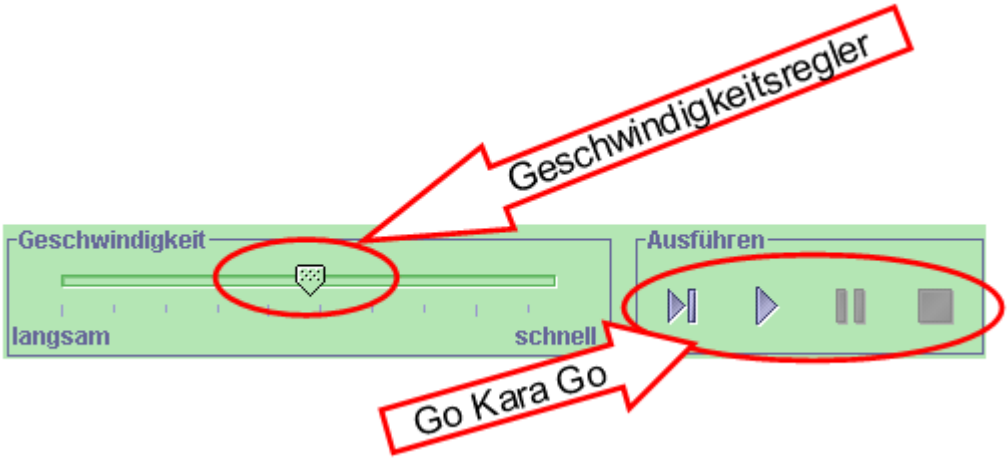


Abb. 3.11: Über dieses Menü kann Kara mit dem aktuellen Programm gestartet werden. Mit dem linken Symbol unter Ausführen kann man sich Schritt für Schritt durch das Programm klicken. Das ist besonders bei der Fehlersuche hilfreich.

- Lasse Kara in den folgenden Welten laufen und teste, ob dein Programm in diesen Situationen richtig läuft.

Welt 1: 

Welt2: 

Welt3: 

Speichern der Welten und Programmen

Möchtest du Welten und Programme speichern, so kannst du das im jeweiligen Fenster am oberen Rand über das folgende Menü machen:

Abb. 3.12: Menü zum Verwalten von Welten und Programmen.

 Erstellt neue, leere Welt oder neue, leere Programme.

 Öffnet eine bestehende Welt oder ein bestehendes Programm.

 Speichern einer Welt oder eines Programms.

 Zum Speichen einer Welt oder eines Programms unter einem neuen Namen.

Nun wollen wir das bestehende Programm speichern, bevor wir ein neues Programm schreiben.

- Klicke auf das speichern Symbol:  .

Es öffnet sich ein Fenster.

- Gib einen Namen ein und speichere das Programm ab.

Damit ist die Aufgabe 3.2 abgeschlossen.

Wissenssicherung 3.3

Übertrage den Automaten von Abbildung 3.4 ins Kara-Programm. Dieser Automat leitet Kara geradeaus bis zum nächsten Baumund lässt ihn unterwegs alle Blätter aufnehmen

Hinweis: Benutze einen neuen Programmeditor  .

Teste dein Programm anschliessend in den folgenden Welten:

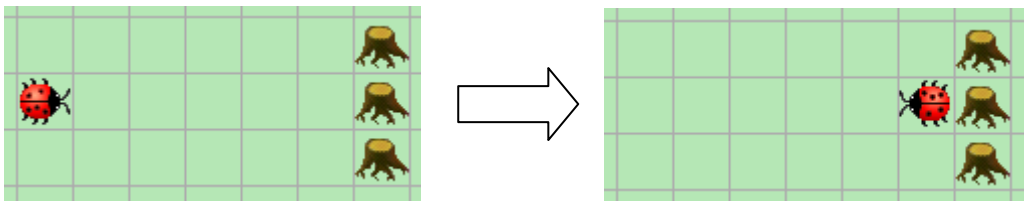
Welt 1: 

Welt 2: 

Welt 3: 

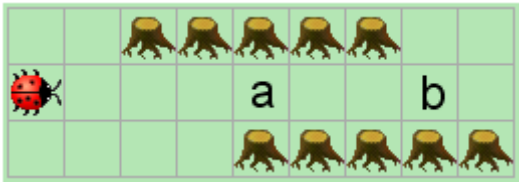
Aufgabe 3.4

Schreibe ein Programm, das Kara geradeaus bis zum nächsten Baum führt. Dort angekommen soll er sich um 180° drehen.



Aufgabe 3.5 a

Kara sucht den Eingang eines geraden Tunnels (Feld a). Schreibe ein Programm, das ihn auf dem ersten Feld im Tunnelinnern anhalten lässt. Aber Achtung: manche Tunnels haben zunächst eine einseitige Wand, manche links, manche rechts.



Teste dein Programm in unterschiedlichen Welten. Bist du sicher, dass Kara die Aufgabe in allen Welten mit einem Tunnel vor sich lösen kann?

Aufgabe 3.5 b

Kara will den Ausgang des Tunnels finden (Feld b). Dazu muss er zunächst den Tunnel durchqueren. Schreibe ein Programm, das ihn auf dem ersten Feld nach dem Tunnel anhalten lässt - er soll nicht bis zum Ende der Galerie laufen!

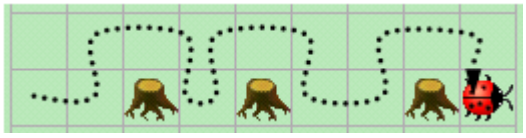
Hinweis: Die Lösung erfordert zusätzlich zum Stoppzustand zwei Zustände!

Kapiteltest

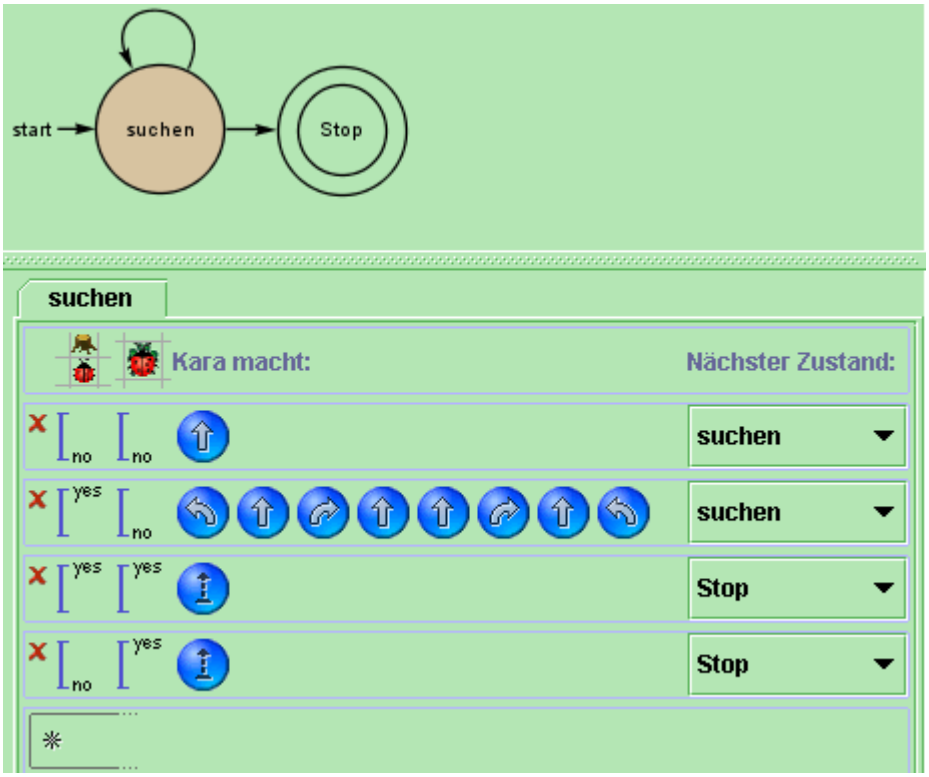
Wenn dir klar ist, wie die Automaten aus den Aufgaben funktionieren, kannst du dich bei der Lehrperson zum Kapiteltest melden.

Lösungen Kapitel 3

Wissenssicherung 3.1

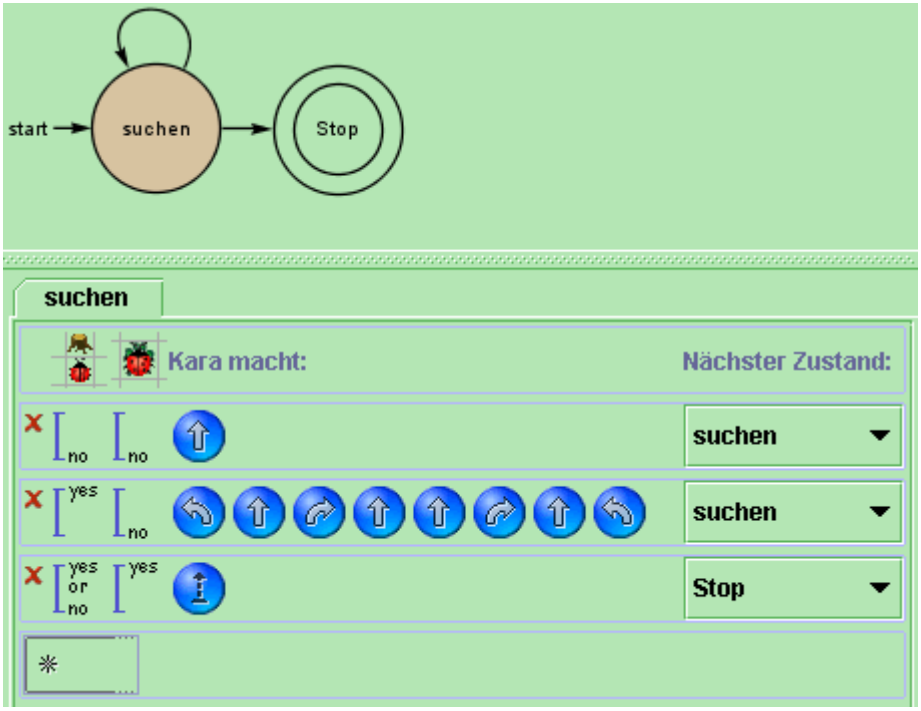


Aufgabe 3.2 a



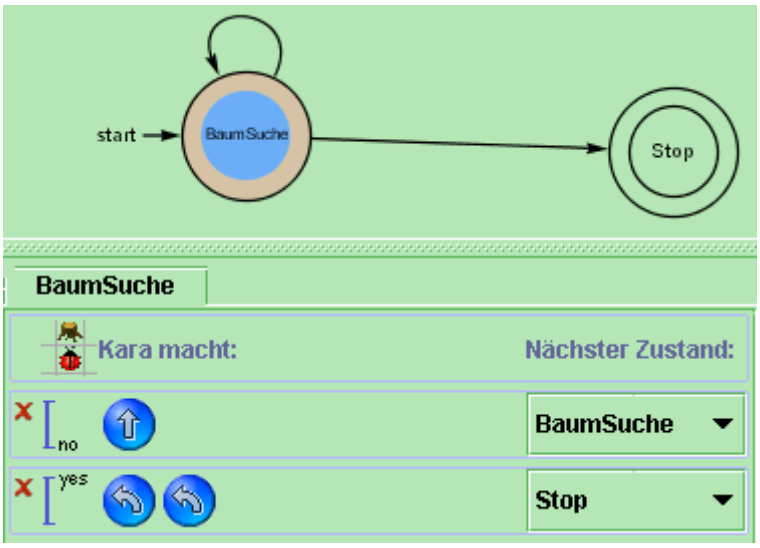
In den beiden letzten Übergängen unterscheiden dich die Aktionen nicht, die ausgeführt werden sollen. Immer wenn Kara ein Kleeblatt gefunden hat, soll er es aufnehmen und die Arbeit abschliessen, egal ob er vor einem Baum steht oder nicht. Darum können wir die beiden letzten Übergänge zusammenlegen, wie es auf der nächsten Seite in Lösung b zu sehen ist.

Aufgabe 3.2 b

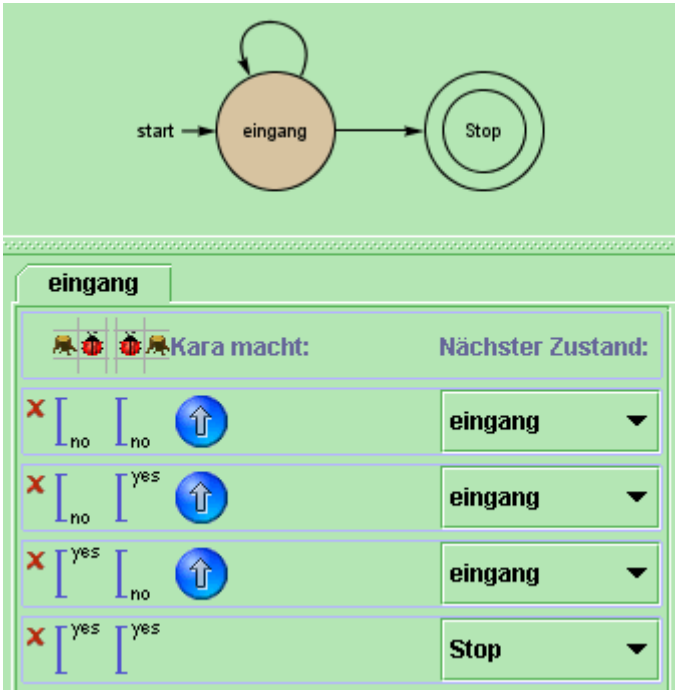


Wissenssicherung 3.3

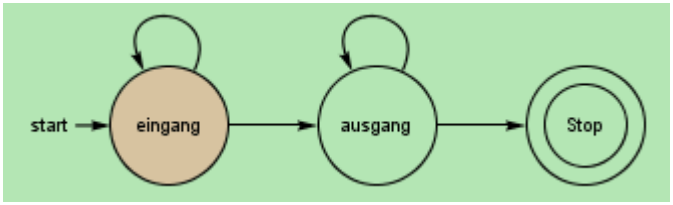
Aufgabe 3.4



Aufgabe 3.5 a



Aufgabe 3.5 b


start
eingang
ausgang
Stop


eingang
Kara macht:
Nächster Zustand:
no
no
eingang
no
yes
eingang
yes
no
eingang
yes
yes
ausgang


ausgang
Kara macht:
Nächster Zustand:
no
no
Stop
no
yes
Stop
yes
no
Stop
yes
yes
ausgang

Kapitel 4 Aufgabensammlung

Übersicht

Worum geht es?

Dieses Kapitel ist nicht mehr als Leitprogramm geschrieben. Du weisst nun, wie Kara programmiert wird und kannst selbständig Aufgaben lösen.

Was tust du?

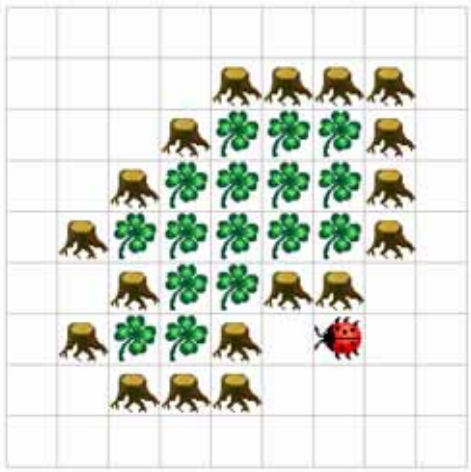
Auf den folgenden Seiten hast du eine Aufgabensammlung. Die Aufgaben sind jeweils mit einem Schwierigkeitsgrad zwischen 1 und 5 markiert, wobei die einfachsten Aufgaben mit 1 gekennzeichnet sind. Du kannst nach Lust und Laune einzelne Aufgaben wählen und lösen. Die Lösungen zu den Aufgaben findest du ab Seite 45.

Lernziele

Nach dem Lösen einiger Aufgaben wirst du dich bald mit Automaten vertraut fühlen.

Kara, der Kleeblatt-Bewacher!

Kara hat sich einen leckeren Vorrat von Kleeblättern angelegt. Dieser ist von Bäumen umringt. Damit sich niemand an seinen Kleeblättern vergreift, beschliesst Kara, diesen Wald zu patrouillieren, ihn immer wieder abzulaufen. So sieht der „Vorratswald“ aus:



Die Aufgabe (Schwierigkeitsgrad: 2 von 5)

Programmiere Kara so, dass er endlos um diesen „Wald“ läuft! Du kannst selbst entscheiden, ob du ihn links- oder rechtsherum laufen lässt.

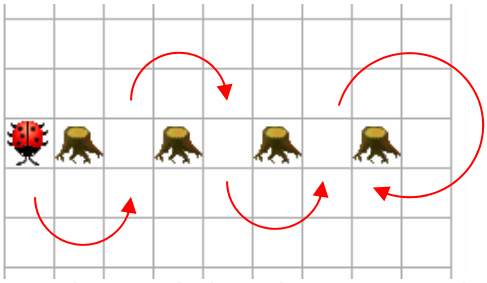
Hinweis: Ein Zustand reicht für die Lösung der Aufgabe aus. Versuche, mit möglichst wenig Sensoren auszukommen, und möglichst wenig Übergänge in deinem Zustand zu haben!

Varianten

- Falls dein Programm mehr als einen Zustand hat: Schreibe eine „optimierte“ Fassung, die nur einen Zustand hat!
- Erweitere dein Programm so, dass Kara zuerst geradeaus bis zum nächsten Baum läuft und erst dann mit dem Patrouillieren beginnt. Dazu musst du ihn vor Programmbeginn allerdings so setzen, dass er geradeaus laufend überhaupt einen Baum findet!

Kara, der Slalom-Skater

Inline-Skates faszinieren Kara! Er möchte üben, zwischen Bäumen Slalom zu „fahren“:



Starten mit einer Linksdrehung, dann zwischen den Bäumen elegant in eine Rechtsdrehung übergehen, dann zwischen den nächsten Bäumen wiederum in einer Linksdrehung… der Anfang des Slaloms ist eingezeichnet; so weiter ad infinitum! Oder bis Kara schwindlig wird!

Die Aufgabe (Schwierigkeitsgrad: 2 von 5)

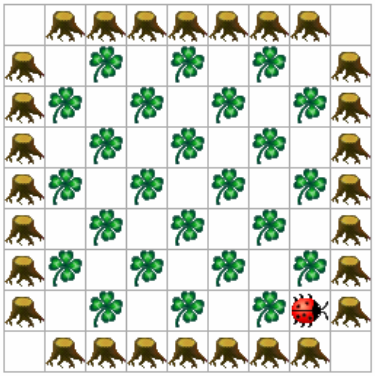
Programmiere Kara so, dass er diesen Slalom fahren kann. Wie lang der Parcours ist (wie viele Bäume nebeneinander stehen), weiss Kara zu Beginn natürlich nicht. Auch soll es ihm egal sein, ob die Bäume horizontal oder vertikal nebeneinander stehen!

Zusatzfrage

Deine Lösung hat mindestens zwei Zustände. Warum kann es mit weniger nicht gehen?

Kara, der Parkettleger

Kara möchte sich als Parkettleger betätigen. Er will ein schachbrettartiges Muster legen:



Die Aufgabe (Schwierigkeitsgrad: 2 von 5)

Programmiere Kara so, dass er ein solches Muster innerhalb der Bäume erzeugt. Es gibt zwei grundsätzlich verschiedene Möglichkeiten, ein solches Muster zu erzeugen. Kannst du dir denken, welche zwei Varianten möglich sind?

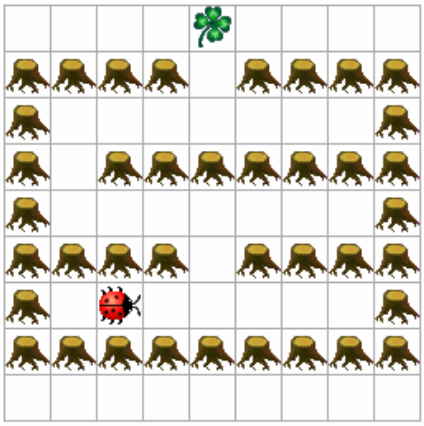
Hinweis: Wähle die Weltgrösse so, dass dein Programm möglichst einfach wird. Je nach Verfahren sind die Anforderungen an die Weltgrösse unterschiedlich. Hilft es deinem Programm, wenn die Länge und Breite der Welt ungerade oder gerade sind? Oder hilft es zu wissen, dass Länge und Breite gleich gross sind?

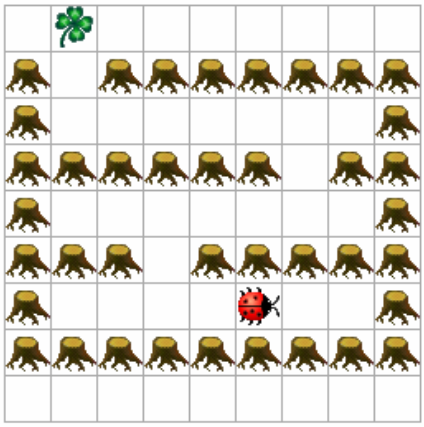
Zusatzfrage

Wie gross ist die kleinste Welt, die dein Programm bearbeiten kann?

Einfaches Labyrinth – Kleeblatt-Suche

Kara sitzt in einem Labyrinth fest. Er möchte raus, denn beim Ausgang des Labyrinths wartet ein leckeres Kleeblatt auf ihn! Zwei Beispiele von Labyrinth-Welten:





Jede horizontale Baumreihe ausser der untersten hat genau einen Ausgang. Diesen muss Kara jeweils finden. Hinter dem letzten Ausgang wartet das Kleeblatt auf ihn!

Die Aufgabe (Schwierigkeitsgrad: 3 von 5)

Programmiere Kara so, dass er das Kleeblatt findet. Er soll das Kleeblatt aufnehmen; damit soll das Programm enden. Die Startposition und Startrichtung in der untersten Labyrinthzeile kannst du frei wählen. Du musst aber dein Programm daran anpassen.

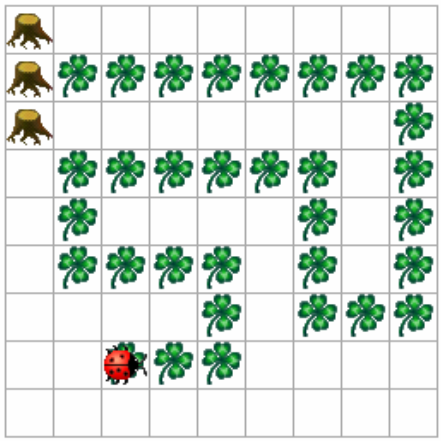
Hinweis: Teste Dein Programm an mehreren verschiedenen Labyrinthen! Sonst läufst Du Gefahr, dass Dein Programm nur für genau einen Spezialfall von Labyrinth funktioniert. Das ist beim Programmieren immer eine sehr grosse Gefahr und kann leicht zu Problemen beim Einsatz eines Programms führen!

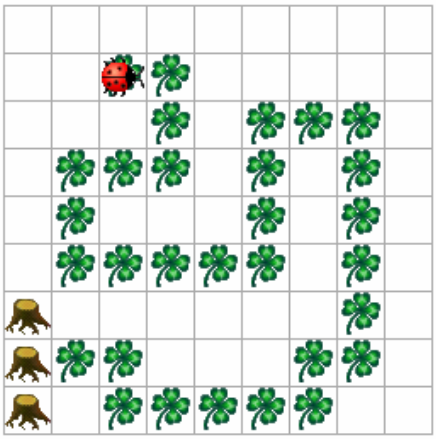
Zusatzfragen

- Wie verhält sich dein Programm, wenn es mehr als ein Loch in einer Baumreihe hat? Funktioniert es immer noch? Warum oder warum nicht?
- Falls dein Programm drei Zustände hat: Versuche, mit zwei Zuständen auszukommen. Ginge es auch mit einem einzigen Zustand?

PacMan – oder wie verfolge ich eine Spur?

Kara möchte das uralte PC-Spiel PacMan spielen – in einer vereinfachten Version. Er soll eine Spur von Kleeblättern „auffressen“. Er weiss, dass diese Spur nie entlang eines Baumes geht – sie endet an einem Baum! So sehen „Spur-Welten“ aus:





Die Aufgabe (Schwierigkeitsgrad: 3 von 5)

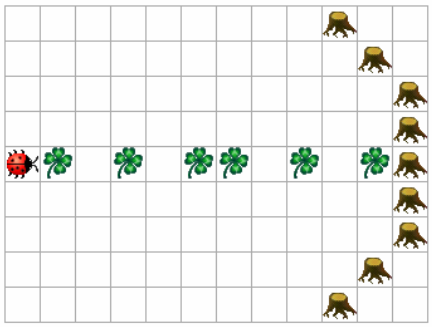
Programmiere Kara so, dass er die Spur von Kleeblättern „auffrisst“! Da du weisst, dass die Spur nie entlang eines Baums geht, kann das Programm beendet werden, sobald Kara auf einem Kleeblatt vor einem Baum steht. Du kannst selbst bestimmen, ob du auf einem Kleeblatt oder davor starten willst.

Hinweis I: Die Lösung ist trickreich! Überlege dir zuerst auf Papier genau, in welchen Situationen Kara sich finden kann, und was er tun muss!

Hinweis II: Teste dein Programm an mehreren verschiedenen Spuren! Sonst läufst du Gefahr, dass dein Programm nur für genau einen Spezialfall von Spur funktioniert. Das ist beim Programmieren immer eine sehr grosse Gefahr und kann leicht zu Problemen beim Einsatz eines Programms führen!

Kara und die Ausserirdischen – Muster in der Welt (1)

Kara glaubt an Ausserirdische. Er ist überzeugt, dass bestimmte Muster von Kleeblättern in seiner Welt auf ihre baldige Ankunft hinweisen! Nur weiss er nicht so recht, welches Muster er erwarten soll. Er stellt sich vor, dass es eines von drei Mustern ist. Betrachten wir als erstes folgende Welt:



Erste Aufgabe (Schwierigkeitsgrad: 3 von 5)

Programmiere Kara so, dass er prüft, ob in der Zeile, in der er steht, folgendes Muster vorkommt:

Er soll solange das Muster suchen, bis er an einem Baum ankommt. Dann gibt er die Suche auf. Sobald er das Muster findet, soll er seine Freude darüber signalisieren, indem er sich links dreht und in die darüber liegende Zeile begibt. Danach kann er das Programm beenden. Findet er es nicht, geht er enttäuscht nach rechts in die darunter liegende Zeile.

Hinweis I: Für ein Muster, das drei Felder „lang“ ist, brauchst du drei Zustände. Überlege dir zuerst auf Papier, wie du diese Zustände als Gedächtnis einsetzen kannst!

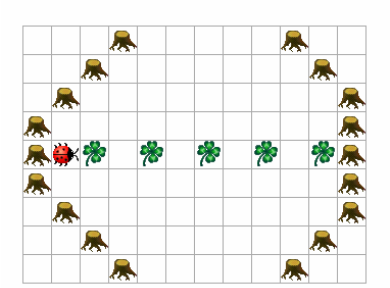
Hinweis II: Testen dein Programm an mehreren verschiedenen Bildern! Teste, ob es funktioniert, wenn das Muster vorkommt, wenn es nicht vorkommt, wenn es nur teilweise vorkommt und an dem Baum „unterbrochen“ wird. Dein Programm muss in allen Fällen die korrekte Antwort geben!

Variante

Probiere ein anderes Muster aus! Gebe dir eines vor, und versuche, es zu finden. Je länger das Muster, desto mehr Zustände brauchst du! Warum?

Kara und die Ausserirdischen – Muster in der Welt (2)

Kara meint, ein anderes Muster könnte ihm auch einen Hinweis geben. Allerdings müsste dieses Muster beliebig oft wiederholt sein in einer Zeile. Die Welt sieht wie folgt aus:



Zweite Aufgabe (Schwierigkeitsgrad: 3 von 5)

Programmiere Kara so, dass er prüft, ob in der Zeile, in der er steht, folgendes Muster „aneinandergereiht“ ist:

Anders gesagt, die Zeile soll (wie im Bild) abwechslungsweise Felder ohne und mit Kleeblättern aufweisen. Sie muss ohne Kleeblatt beginnen und mit einem Kleeblatt enden – dann ist das Muster eine gewisse Anzahl mal aufgetreten! Im obigen Bild ist es fünfmal vertreten.

Kommt Kara zum Schluss, dass die Zeile das richtige Muster aufweist, soll er seine Freude darüber signalisieren, indem er sich links dreht und in die darüber liegende Zeile begibt. Findet er es nicht, geht er enttäuscht nach rechts in die darunter liegende Zeile.

Hinweis: Teste dein Programm an mehreren verschiedenen Bildern! Teste, ob es funktioniert, wenn die Bildzeile dem beschriebenen Aufbau nicht genügt. Dein Programm muss in jedem Fall die korrekte Antwort geben!

Variante

Probiere andere aneinander gereihte Muster aus! Gebe dir ein Muster vor, und versuche, es zu finden.

Kara und die Ausserirdischen – Muster in der Welt (3)

Kara meint, noch ein weiteres Muster könnte ihm einen Hinweis geben. Das Muster wäre: zuerst eine beliebige Anzahl Felder ohne Kleeblätter, dann die gleiche Anzahl Felder mit Kleeblättern:

Dritte Aufgabe (Schwierigkeitsgrad: 3 von 5)

Kara hat allerdings seine Zweifel, ob seine Fähigkeiten ausreichen, ein derartiges Muster zu erkennen. Vor allem weiss er nicht von vornherein, wie lange die Zeile ist. Vielleicht lebt er ja in einer viel viel längeren Welt! Auch will er die Welt nicht verändern, denn sonst verärgert er womöglich die Ausserirdischen.

Was meinst du: Reichen Kara's Fähigkeiten aus, zu prüfen, ob die Zeile ein solches Muster enthält? Gehe wie folgt vor:

1. Überlege dir zunächst auf Papier eine Lösung für den Fall, dass die Zeile genau 4 Felder lang ist. Kein Problem, oder?

2. Überlege dir nun, wie eine Lösung aussieht, wenn du weisst, dass die Zeile maximal 4 Felder aufweist!

   Hinweis: Die Lösung braucht 4 Zustände.

3. Programmiere deine Lösung von (3)! Auch hier gilt: Teste dein Programm an möglichst vielen Mustern!!

   Hinweis: Schwierigkeitsgrad: 3-4.

4. Versuche, daraus abzuleiten, wo das Problem mit ganz langen Zeilen liegt!

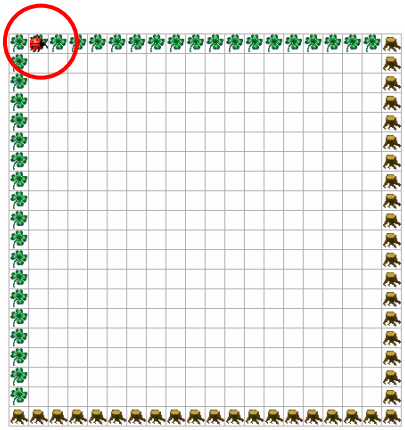
Tja, wenn sich die Ausserirdischen mit einem derartigen Muster anmelden... Pech gewesen!

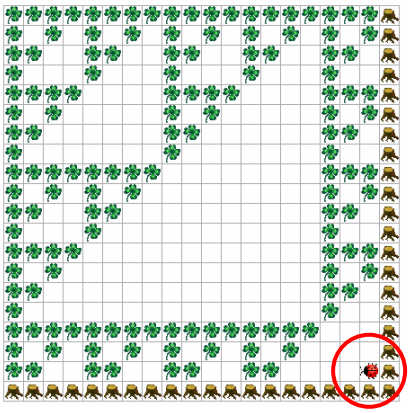
Freiwillige Zusatzaufgabe (Schwierigkeitsgrad: 5 von 5)

Wenn Kara erlaubt wird, die Welt zu verändern, so kann er die Aufgabe lösen! Er könnte zum Beispiel die Zeile ober- oder unterhalb der „Musterzeile" als Gedächtnis, als „Speicher" benutzen. Aber Achtung: die Lösung braucht 5 Zustände und ist ziemlich aufwendig!

Kara, der Mathematiker

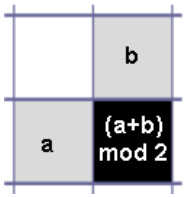
Kara möchte das berühmte Pascal-Dreieck zeichnen, bestehend aus Kleeblättern:





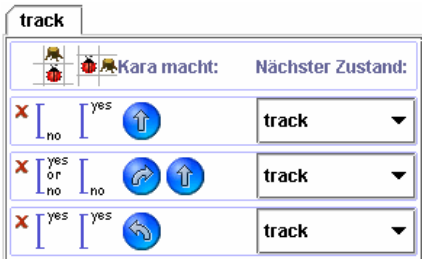
Die Aufgabe (Schwierigkeitsgrad: 4 von 5)

Programmiere Kara so, dass er ein binäres Pascal-Dreieck zeichnet. Das heisst, jede Zahl wird „modulo 2“ abgebildet: eine gerade Zahl wird als freies Feld dargestellt, eine ungerade Zahl als Feld mit einem Kleeblatt drauf. Der Einfachheit halber kippen wir das Dreieck in die linke obere Ecke der Welt. Die obige Abbildung zeigt die Ausgangslage für diese Aufgabe und das von Kara gezeichnete Pascal-Dreieck. Wie die inneren Felder berechnet müssen, zeigt folgende Abbildung:



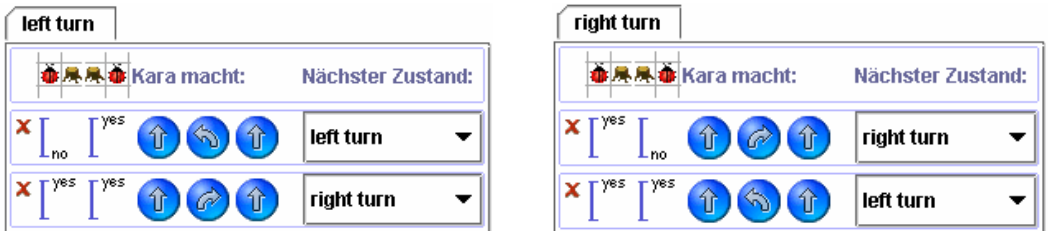
Hinweis: Denke daran: die Zustände sind Kara’s Gedächtnis! Diese Tatsache musst du bei dieser Aufgabe ausnützen. Mit weniger als drei Zuständen dürftest du nicht auskommen.

Lösung: Kara, der Kleeblatt-Bewacher! (mit Rechtsdrehung)



Das ist alles! In jeder Situation stellt das Programm sicher, dass nach der Ausführung der Befehle entweder rechts von Kara oder rechts hinter Kara wieder ein Baum ist. So folgt Kara endlos der „Wand“ aus Bäumen…

Lösung: Kara, der Slalom-Skater



Start: Wenn Kara wie in der Abbildung steht, in „left turn“. Schaut er nach oben, hat also den Baum rechts von sich, dann in „right turn“.

Lösung Zusatzfrage

Die Antwort auf die Zusatzfrage steckt eigentlich schon in der Aufgabenstellung: „*Starten mit einer Linksdrehung, dann zwischen den Bäumen elegant in eine Rechtsdrehung übergehen, dann zwischen den nächsten Bäumen wiederum in einer Linksdrehung…*“. Die beiden Zustände sind Kara’s Gedächtnis. Ohne sie kann er sich nicht merken, ob er *zwischen* zwei Bäumen eine Links- oder Rechtsdrehung vollziehen muss!

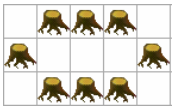
Lösung: Kara, der Parkettleger

1: Kara läuft horizontal (oder vertikal) hin und her

Die Welt muss horizontal 2+1+n*2 (n>=1) Felder haben: zwei für die Bäume, 1 für die

Startposition und jeweils zwei für jeden Zustandsübergang. Auf diese Weise kommt die Lösung mit nur zwei Zuständen aus. Vertikal muss mindestens eine Zeile zwischen den Bäumen frei sein.

Weltgrösse: Somit sieht die kleinstmögliche Welt, 5 auf 3 Felder gross, wie folgt aus:

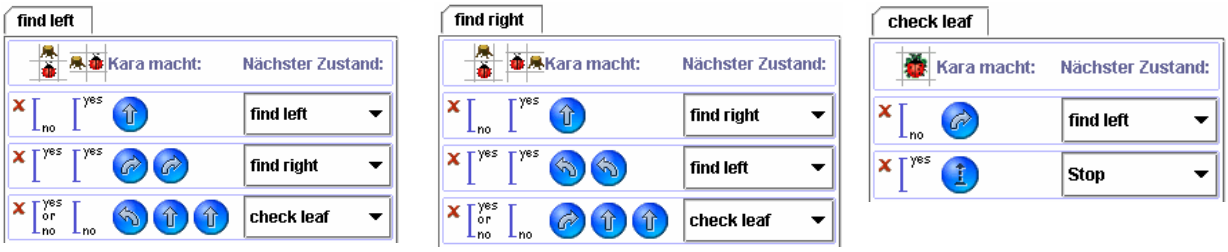


2: Kara läuft diagonal hin und her

Start: Im linken unteren Ecken der Welt, nach rechts schauend, im Zustand „no leaves“.
Weltgrösse: Die kleinste Welt ist 4x4 Felder gross.

Lösung: Einfaches Labyrinth – Kleeblatt-Suche

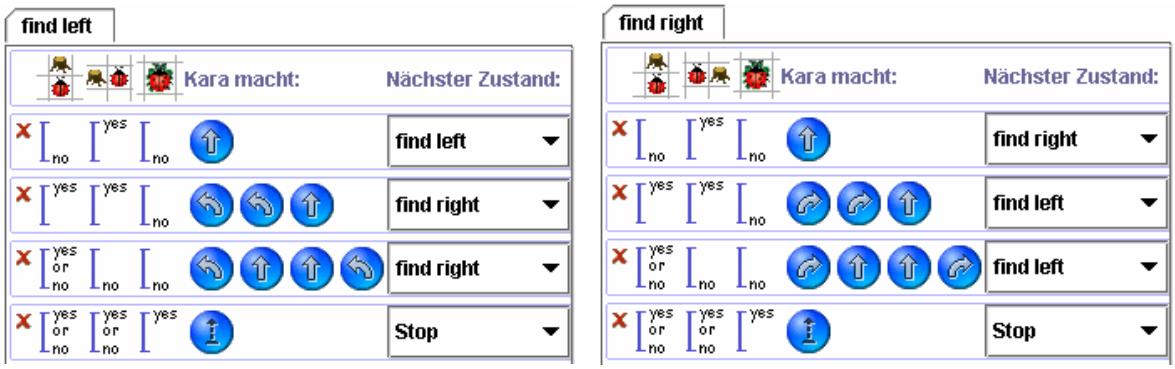
Lösung mit drei Zuständen



Start: „find left“, wenn Kara nach rechts schaut; „find right“, wenn er nach links schaut.

Anmerkung: Es ist egal, ob in „check leaf“ bei „leaf on ground? no“ nach „find left“ oder „find right“ gegangen wird. Kara muss nur in der entsprechenden Richtung gedreht werden.

Optimierte Lösung mit zwei Zuständen



Start: „find left“, wenn Kara nach rechts schaut; „find right“, wenn er nach links schaut.

Anmerkung: Man könnte noch in einem der beiden Zustände den Sensor „leaf on ground?“ einsparen.

Zu den Zusatzfragen

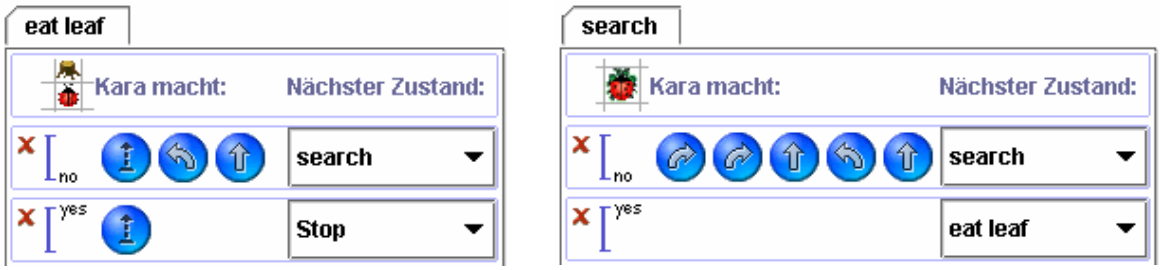
Löcher in den horizontalen Baumreihen stören das Programm nicht – solange die oberste Baumreihe nur genau bei dem Kleeblatt ein Loch hat! Oder aber die Löcher sind an das Programm angepasst so positioniert, dass sie den Programmablauf nicht stören!

Mit einem Zustand kann es nicht gehen. Kara kann mit nur einem Zustand bei einem „Loch“ in der Baumreihe zu seiner Linken oder Rechten nicht wissen, ob er daran schon mal vorbeigekommen ist oder nicht, ob er hindurch soll oder nicht!

Wenn hingegen die Aufgabenstellung vereinfacht würde: die Löcher sind immer am Ende einer Baumreihe, dann würde ein Zustand ausreichen!

Lösung: PacMan – oder wie verfolge ich eine Spur?

Lösung mit zwei Zuständen



Start: in „eat leaf“, wenn Kara auf Kleeblatt startet; in „search“, wenn er davor steht.

Lösung mit nur einem Zustand

Bemerkung: „Optimiert“ ist hier nur die Anzahl Zustände – einer wurde eingespart. Sie lassen sich problemlos kombinieren, weil sie ganz andere Situationen abdecken. Die Anzahl Befehle hat sich nicht geändert.

Eine weitere, um einiges schwierigere Aufgabe wäre es, Spuren zu „fressen“, die ganzen Baumreihen entlang gehen dürfen. Vielleicht nimmt ja jemand die Herausforderung an?

Lösung: Kara und die Ausserirdischen – Muster in der Welt (1)

Mit den Zuständen merkt sich Kara, welchen Teil des Musters er bereits erkannt hat. In „start“ fängt das Programm an. In „0“ weiss Kara, dass er bereits ein leeres Feld hinter sich hat. In „01“ weiss er, dass er ein leeres Feld und eines mit Kleeblatt hinter sich hat. Liegt daher in „01“ kein Kleeblatt auf dem Feld, weiss er dass er das Muster erkannt hat!

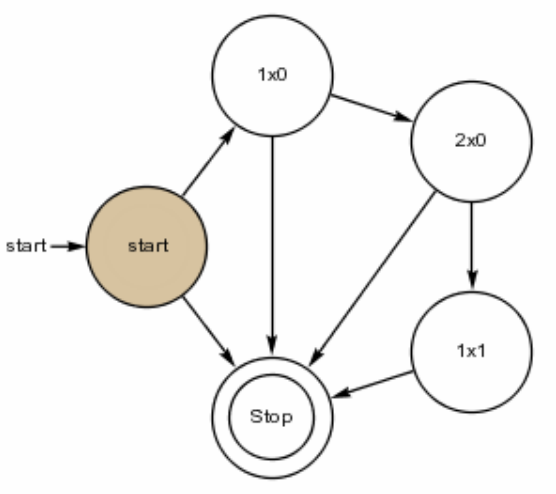
Lösung: Kara und die Ausserirdischen – Muster in der Welt (2)

Dieses Programm funktioniert ähnlich wie das der ersten Aufgabe. Im Zustand „0“ weiss Kara, dass ein leeres Feld hinter ihm liegt. Steht er auf einem Kleeblatt vor einem Baum, erfüllt die Zeile die Suchbedingung.

Lösung: Kara und die Ausserirdischen – Muster in der Welt (3)

Zu 1. Die Lösung reduziert sich in diesem Fall auf „prüfe Muster“, leicht abgewandelt von der ersten Such-Aufgabe.

Zu 2, 3. Mit einer oberen Schranke gibt es eine relativ elegant, quasi symmetrische Lösung. Der Einfachheit halber hier nur die Übersicht:



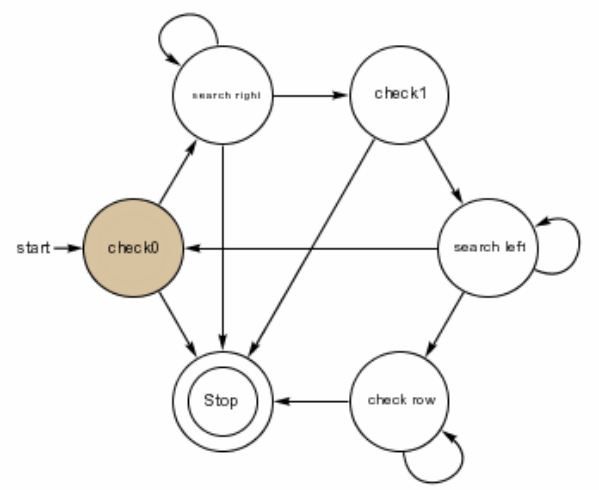
Die Idee ist einfach. Die Zustände merken sich, welchen Teil des Musters sie schon erkannt haben. Sie wissen auch, welcher Teil noch vor ihnen liegen muss. Diese Lösung kann leicht verallgemeinert werden für beliebige obere Schranken.

Zu 4. Es ist aber keine praktikable Lösung. Da Kara kein Gedächtnis in Form eines internen Zählers (Variable) hat, müssen Zustände die Rolle des Gedächtnisses übernehmen. Für 6 Felder kann ein Automat mit 6 Zuständen erstellt werden. Es gilt: für *n* Felder kann ein Automat mit *n* Zuständen erstellt werden.

Ein endlicher Automat kann nur reguläre Ausdrücke erkennen; das Muster „*n* leere Felder, *n* Felder mit Kleeblättern“, wobei *n* mindestens eins sein muss, ist kein regulärer Ausdruck – zumindest, wenn n gegen unendlich gehen darf. Hier könnte die ganze Theorie von der Mächtigkeit regulärer Ausdrücke und kontextfreier Grammatiken gezeigt werden…

Lösung der Zusatzaufgabe

Darf Kara die Welt verändern, so kann er die Aufgabe lösen. Die Welt dient ihm dann als Gedächtnis, als „Speicher“. Denn wäre die Welt unendlich lang, so könnte Kara jede Berechnung der Welt anstellen, weil er dann eine Turingmaschine realisieren könnte… Hier geht’s aber etwas einfacher! Der Einfachheit halber hier die Lösung [pattern0n1n.kara] mit der Übersicht:



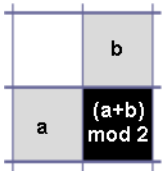
Start: Kara muss auf dem leeren Feld starten, dass vor dem ersten Feld mit Kleeblatt liegt, und muss in dessen Richtung schauen.

Die Idee ist folgende. In der Zeile oberhalb der Musterzeile markiert Kara die Felder, die dem Muster entsprechen. Zu Beginn erwartet er ein leeres Feld. Da er dieses hat, markiert er das Feld oberhalb mit einem Kleeblatt. Dann läuft er nach rechts auf der Suche nach dem ersten nicht markierten Feld. Unterhalb von diesem muss ein Kleeblatt sein. Ist dem so, markiert er das Feld oberhalb, läuft nach links auf der Suche nach einem unmarkierten Feld, prüft das Feld unter diesem Feld…

Die Zeile erfüllt das Muster genau dann, wenn bei Programmende die obere Zeile leer ist!

Lösung: Kara, der Mathematiker

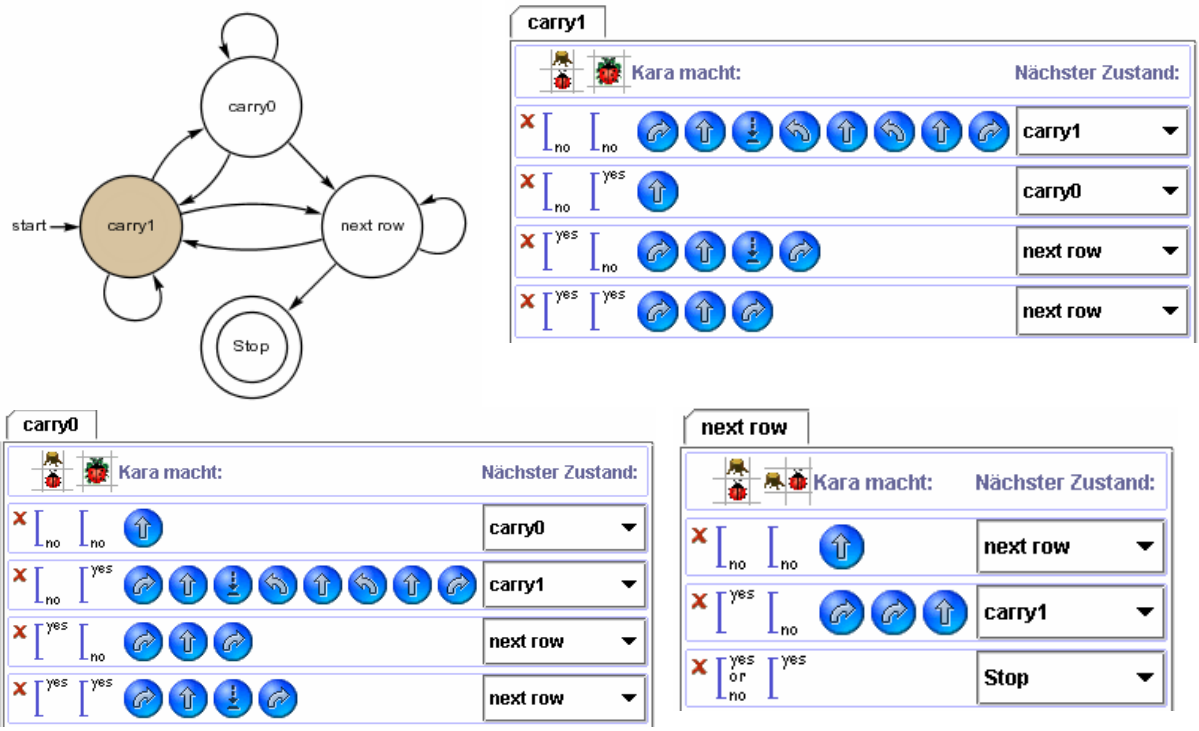
Die Berechnung der Werte im Pascal-Dreieck, das heisst, das Plazieren von Kleeblättern durch Kara, ist im Prinzip einfach: Kara berechnet das Pascal-Dreieck zeilenweise von links nach rechts. Erreicht Kara bei seiner Berechnung das Ende einer Zeile, markiert durch einen Baum, so läuft er im Zustand `next row` an den Anfang der nächsten Zeile.



Wie berechnet Kara die Werte in den einzelnen Zellen des Pascal-Dreiecks? Kara kann über den Sensor „auf Kleeblatt?“ feststellen, ob er sich auf einem Feld mit einem Kleeblatt (also einer ungeraden Zahl) befindet oder nicht (das Feld **b** in obiger Abbildung links). Dann muss er noch wissen, ob sich auf dem Feld unmittelbar rechts hinter ihm (das Feld **a** in obiger Abbildung; die eingekreisten Felder rechts) ein Kleeblatt befindet oder nicht. Diese Information wird dadurch abgespeichert, dass sich Kara in einem der beiden folgenden Zustände befindet:

- `carry0`: Auf dem Feld hinten rechts von Kara befindet sich kein Kleeblatt (also eine gerade Zahl)
- `carry1`: Auf dem Feld hinten rechts von Kara befindet sich ein Kleeblatt (also eine ungerade Zahl)

Befindet sich Kara im Zustand `carry0` und steht auf einem Feld mit keinem Kleeblatt, so bleibt das Feld rechts von Kara leer und Kara macht einen Schritt vorwärts. Falls er auf einem Feld mit einem Kleeblatt steht, so resultiert für das Feld rechts von Kara eine ungerade Zahl. Kara legt dort ein Kleeblatt hin und macht dann einen Schritt vorwärts. Entsprechend verhält sich Kara im Zustand `carry1`. Folgende Abbildung zeigt das ganze Programm.



Anhang 1: Kapitel-Test für den Tutor

Kapiteltest 1

Aufgabe 1

Beschreibe in eigenen Worten, was du unter Sensoren verstehst. Wofür werden sie eingesetzt?

Aufgabe 2

Gegeben ist die folgende Welt:

Kara führt die folgenden Bewegungen aus:

Fülle untenstehende Tabelle aus. Jeweils wieder ein Kreuz für nicht erfüllt und ein Häklein für erfüllt.

Schritte	0	1	2	3	4	5	6	7	8
Baum vorne?									
Baum links?									
Baum rechts?									
Pilz vorne?									
Kleeblatt unten?									

Kapiteltest 1 Lösung

Aufgabe 1 (K2)

Über die Sensoren kann Kara Informationen über seine Umgebung erfassen. So kann er zum Beispiel feststellen, ob auf dem Feld direkt vor ihm ein Baumstrunk steht oder nicht.

Wir benötigen diese Informationen, damit wir Kara allgemein programmieren können. Das heisst, dass Kara fähig sein muss seine Aufgaben in unterschiedlichen, aber gleich strukturierten Welten zu lösen. Kara soll zum Beispiel alle Kleeblätter bis zu einem Baumstunk aufnehmen können, egal wie viele Kleeblätter herum liegen und wie sie angeordnet sind.

Aufgabe 2 (K2)

Schritte	0	1	2	3	4	5	6	7	8
Baum vorne?	✔	×	×	×	×	×	×	×	×
Baum links?	✔	✔	×	×	×	×	×	×	✔
Baum rechts?	×	×	✔	×	×	✔	×	×	✔
Pilz vorne?	×	×	✔	✔	✔	✔	✔	✔	✔
Kleeblatt unten?	×	×	✔	×	×	✔	✔	✔	×

Kapiteltest 2

Aufgabe 1

Erkläre in Deinen eigenen Worten die Begriffe Zustand, Übergang, Input, Aktionen und Startzustand.

Aufgabe 2

Notiere 3 wenn..., dann Aussagen zum Videorecorder-Automat aus Aufgabe 2.3.

Kapiteltest 2 Lösung

Aufgabe 1 (K2)

- Zustand: Ein Automat befindet sich immer in einem Zustand und wartet auf neuen Input. Die Zustände stellen das Gedächtnis des Automaten dar. Er kann sich zum Beispiel merken, wie viel Geld ein Käufer bereits in den Getränkeautomaten eingeworfen hat.

- Input: Ein Automat reagiert nur auf neuen Input von aussen. Oft erfolgt dieser Input über Sensoren.

- Aktionen: Die Aktionen sind die der Automat gibt, wie auf gewissen Input zu reagieren ist. Zum Beispiel soll der Getränkeautomat die Aktion „Ausgabe einer Cola“ ausführen, wenn der Käufer auf den Knopf Cola drückt. Natürlich dürfen nur jene Aktionen ausgeführte werden, die auf Übergängen aus dem aktuellen Zustand stehen. So muss der Getränkeautomat im Zustand „3.- Kredit“ sein, damit er eine Cola ausspucken darf.

- Übergang: Übergänge sind die Verbindungen zwischen den Zuständen. Auf den Übergängen sind Input und Aktionen vermerkt. Der Input ist die Bedienung, die erfüllt sein muss, damit ein bestimmter Übergang genommen werden darf. Der Automat befindet sich anschliessend im Zustand am Ende des Übergangs.

- Startzustand: Nach dem Start befindet sich ein Automat in diesem Zustand.

Aufgabe 2 (K2)

Es gibt viele verschiedene Möglichkeiten. Hier 3 Beispiele:

- Wenn sich der Automat im Zustand „warten“ befindet und den Input „spielen“ erhält, dann soll er in den Zustand „spielen“ wechseln und den Video abspielen.

- Wenn sich der Automat im Zustand „spielen“ befindet und den Input „spulen“ erhält, dann soll er in den Zustand „spulen“ wechseln und den Video mit Bild spulen.

- Wenn sich der Automat im Zustand „warten“ befindet und den Input „spulen“ erhält, dann soll er in den Zustand „spulen“ wechseln und den Video ohne Bild spulen.

Kapiteltest 3

Aufgabe 1 (K3)

Zeichne Deinen Automat aus Aufgabe 3.5 b auf ein Blatt Papier oder rufe die Lehrperson direkt an deinen Computer. Erkläre wie dein Automat Schritt für Schritt die folgende Aufgabe löst:

Aufgabe 2 (K3)

Warum kann in Aufgabe 3.5 b ein Zustand nicht ausreichen?

Kapiteltest 3 Lösung

Aufgabe 1 (K3)

Bei dieser Aufgabe soll der Schüler oder die Schülerin seine/ ihre eigene Lösung selber erklären und anhand des gezeigten Beispiels noch einmal Schritt für Schritt überdenken.

Aufgabe 2 (K3)

Ein Zustand kann nicht ausreichen, da Kara auf die Situation einer einseitigen Wand unterschiedlich reagieren muss, je nachdem ob er sich vor dem Tunnel befindet oder gerade aus dem Tunnel herausgetreten ist. Kara muss sich also merken, ob er noch vor dem Tunnel ist, oder schon im Tunnel. Dafür braucht er zwei Zustände.

Anhang 2: Mediothek, Multimedia

Ich kann kein zusätzliches Material für die Schülerinnen und Schüler empfehlen. Das einzige Buch, das zu Kara existiert ist für Lehrpersonen geschrieben worden.

Anhang 3: Material für die Lernenden

Damit die Schülerinnen und Schüler Kara programmieren können, muss das Softwarepaket Kara auf den Computern installiert werden.

Auf EducETH finden Sie eine genaue Anleitung:

http://www.educeth.ch/informatik/karatojava/download.html

Die Schülerinnen und Schüler brauchen keine zusätzlichen Materialien zum arbeiten.

Für mehr Hintergrundsinformationen zu Kara gibt es das Buch: Programmieren mit Kara. Ein spielerischer Zugang zur Informatik [Reichert 3].

Auf EducETH finden sie auch viele zusätzliche Anregungen und Aufgabestellungen zu Kara:

http://www.educeth.ch/informatik/karatojava

Anhang 4: Von der Autorin benutzte Quellen

Anhang 4a:

Alle Aufgaben, die Kara in diesem Leitprogramm gestellt werden und die Beispiele von alltäglichen Automaten stammen von Raimond Reichert. Zu finden sind sie in den unten aufgelisteten online Dokumente.

Anhang 4b:

[Reichert 1]	Raimond Reichert. *Ein Leitfaden zu Kara*. http://www.educeth.ch/informatik/karatojava/kara/docs/leitfaden.pdf, 28.6.05
[Reichert 2]	Raimond Reichert. *Einführungsvortrag in die Programmierung von Kara.* http://www.educeth.ch/informatik/karatojava/kara/docs/folien_einstieg.pdf, 28.6.05
[Reichert 3]	R. Reichert, J. Nievergelt, W. Hartmann. Programmieren mit Kara. Springer, Berlin. September 2003. ISBN 3540403620